IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF TEXAS
SHERMAN DIVISION

STMicroelectronics, Inc., a Delaware corporation,

Plaintiff,

v.

Broadcom Corporation, a California corporation,

Defendant.

C.A. No. 4:02CV362

COMPLAINT

STMicroelectronics, Inc. ("ST") for its Complaint against defendant Broadcom Corporation ("Broadcom") alleges as follows:

1. This is a patent infringement action to stop Broadcom's unauthorized and infringing sale, offers to sell, use, and importation of products incorporating ST's patented technology. ST is a leader in the design and manufacture of semiconductor devices. ST seeks injunctive relief to stop Broadcom from continuing to infringe ST's valuable patent rights, as well as monetary damages.

2. Plaintiff, ST, is a corporation existing and organized under the laws of Delaware and has its principal place of business at 1310 Electronics Drive, Carrollton, Texas 75006-5039.

3. Defendant, Broadcom, is a corporation existing and organized under the laws of California and has its principal place of business at 16215 Alton Parkway, Irvine, California 92619-7013.

4. This action for patent infringement arises under the Patent Laws of the United States, 35 U.S.C. §§ 1 et seq., and in particular 35 U.S.C. §§ 271, 281, 283, 284 and 285. This Court has jurisdiction over the subject matter of this action under 28 U.S.C. 1338(a).

5. This Court has personal jurisdiction over Broadcom and venue is proper in this Court pursuant to 28 U.S.C. §§ 1391 and 1400.

6. This case involves technology used in semiconductor devices, often referred to as "integrated circuits" or simply "chips." ST and its affiliates own a number of patents relating to semiconductor devices, and have invested millions of dollars in developing the technologies covered in their extensive patent portfolio.

7. United States Patent No. 5,031,092 entitled "Microcomputer With High Density RAM In Separate Isolation Well On Single Chip" (hereinafter "the '092 patent"), was duly and legally issued on July 9, 1991. ST is the owner of the '092 patent and of all rights to sue and recover for infringement thereof. A copy of the '092 patent is attached as Exhibit A.

8. United States Patent No. 5,347,185 entitled "Protection Structure Against Latch-Up In A CMOS Circuit" (hereinafter "the '185 patent"), was duly and legally issued on September 13, 1994. ST is the owner of the '185 patent and of all rights to sue and recover for infringement thereof. A copy of the '185 patent is attached as Exhibit B.

9. United States Patent No. 5,515,225 entitled "Integrated Circuit Protected Against Electrostatic Overvoltages" (hereinafter "the '225 patent"), was duly and legally issued on May 7, 1996. ST is the owner of the '225 patent and of all rights to sue and recover for infringement thereof. A copy of the '225 patent is attached as Exhibit C.

10. United States Patent No. 5,946,261 entitled "Dual-Port Memory" (hereinafter "the '261 patent"), was duly and legally issued on August 31, 1999. ST is the owner of the '261 patent and of all rights to sue and recover for infringement thereof. A copy of the '261 patent is attached as Exhibit D.

11. United States Patent No. 6,025,746 entitled "ESD Protection Circuits" (hereinafter "the '746 patent"), was duly and legally issued on February 15, 2000. ST is the owner of the '746 patent and of all rights to sue and recover for infringement thereof. A copy of the '746 patent is attached as Exhibit E.

12. United States Patent No. 6,087,709 entitled "Method Of Forming An Integrated Circuit Having Spacer After Shallow Trench Fill And Integrated Circuit Formed Thereby" (hereinafter "the '709 patent"), was duly and legally issued on July 11, 2000. ST is the owner of the '709 patent and of all rights to sue and recover for infringement thereof. A copy of the '709 patent is attached as Exhibit F.

13. Broadcom offers to sell, sells, uses, and imports semiconductor devices that are covered by one or more claims of the aforementioned patents.

14. Through its actions including offering to sell, selling, using, and importing the semiconductor devices covered by one or more claims of the aforementioned patents, Broadcom has infringed the aforementioned patents and actively induced others to infringe and contributed to the infringement by others of the aforementioned patents in the United States, including within the jurisdiction of this Court. Upon information and belief, Broadcom does not make the accused products. If contrary to ST's information and belief, Broadcom has in fact made the accused products in the United States, then Broadcom has also infringed the aforementioned patents by its actions in this regard.

15. ST is likely to be irreparably harmed by Broadcom's infringement, inducement of others to infringe, and contributory infringement of the patents in suit. ST has no adequate remedy at law.

WHEREFORE, ST prays for judgment that:

A. United States Patents Nos. 5,031,092, 5,347,185, 5,515,225, 5,946,261, 6,025,746 and 6,087,709 have been infringed, directly, by inducement, and/or contributorily, by Broadcom;

B. Broadcom, its officers, agents, servants and employees, and those persons in active concert and participation with any of them, be permanently enjoined from the direct or contributory infringement of, and from inducing others to infringe United States Patents Nos. 5,031,092, 5,347,185, 5,515,225, 5,946,261, 6,025,746 and 6,087,709;

C. ST be awarded damages sufficient to compensate it for Broadcom's infringement, contributory infringement and inducement of others to infringe, that such damages be increased

to three times the amount found or assessed pursuant to 35 U.S.C. § 284, and that such damages be awarded to ST with prejudgment interest;

D. That this case be declared exceptional pursuant to 35 U.S.C. § 284 and that ST be awarded its attorney fees, costs and expenses in this action; and

E. ST be awarded such other and further relief as the Court may deem just.

ST DEMANDS A TRIAL BY JURY.

Dated: November 7, 2002

Respectfully submitted,

STMicroelectronics, Inc.

By: _____

Clyde M. Siebman
State Bar No. 18341600
SIEBMAN, REYNOLDS, & BURG, L.L.P.
421 N. Crockett
Sherman, Texas 75090
(903) 870-0070
(903) 870-0066 Telefax

Stephen G. Rudisill
State Bar No. 02417049
Edward F. McCormack
State Bar No. 06202555
JENKENS & GILCHRIST
225 West Washington
Suite 2500
Chicago, IL 60606
(312) 425-3909

United States Patent [19]

Edwards et al.

[11] **Patent Number: 5,031,092**

[45] **Date of Patent: * Jul. 9, 1991**

[54] **MICROCOMPUTER WITH HIGH DENSITY RAM IN SEPARATE ISOLATION WELL ON SINGLE CHIP**

[75] Inventors: **Jonathan Edwards**, Bristol; **David L. Waller**, Kent; **Michael D. May**, Bristol, all of England

[73] Assignee: **Inmos Limited**, Bristol, England

[*] Notice: The portion of the term of this patent subsequent to Jul. 14, 2004 has been disclaimed.

[21] Appl. No.: **380,198**

[22] Filed: **Jul. 14, 1989**

Related U.S. Application Data

[60] Division of Ser. No. 938,380, Dec. 9, 1986, which is a continuation of Ser. No. 553,027, Nov. 16, 1983, abandoned.

[30] **Foreign Application Priority Data**

Nov. 26, 1982 [GB] United Kingdom 82/33733

[51] Int. Cl.[5] G06F 13/00; G06F 15/16; G06F 7/48

[52] U.S. Cl. 364/200; 364/232.8

[58] Field of Search ... 364/200 MS File, 900 MS File

[56] **References Cited**

U.S. PATENT DOCUMENTS

3,980,992	9/1976	Levy et al.	364/200
3,993,934	11/1976	Baker et al.	361/400
4,074,293	2/1978	Kravitz	357/40
4,144,563	3/1979	Heuer et al.	364/200
4,153,933	5/1979	Blume, Jr. et al.	364/200
4,191,996	3/1980	Chesley	364/200
4,346,459	0/0000	Sud et al.	365/200
4,349,870	9/1982	Shaw et al.	364/200
4,467,420	8/1984	Murakami et al.	364/200
4,482,950	11/1984	Dshkhunian et al.	364/200
4,491,907	1/1985	Koeppen et al.	364/200
4,546,454	10/1985	Gupta et al.	371/10 X
4,587,542	5/1986	Rao	357/23.6

OTHER PUBLICATIONS

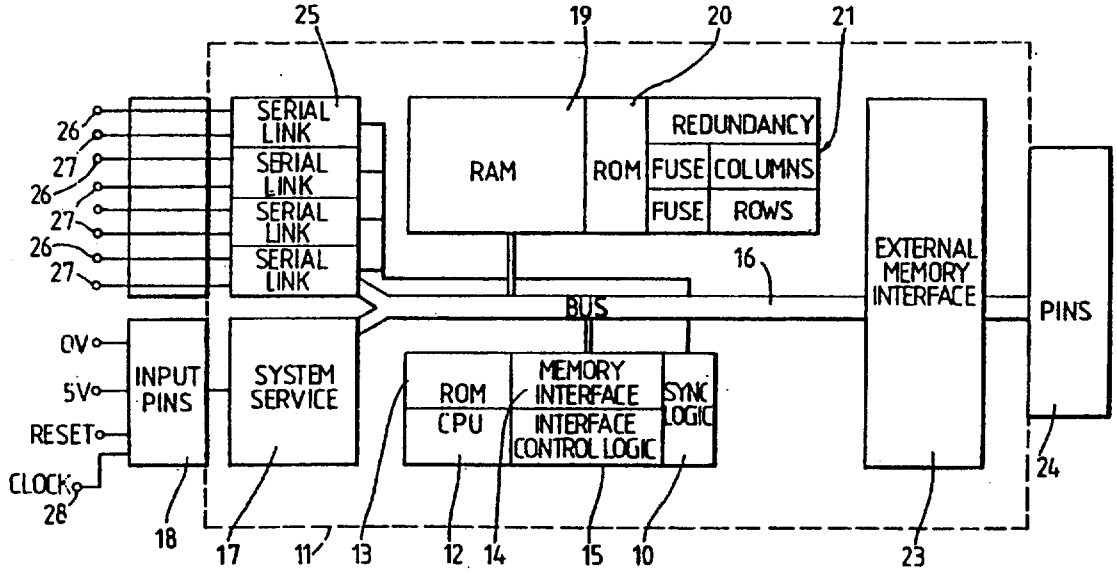
Intel Microcomputer Handbook, Jan. 1983, pp. 16–26.
Electronic Design, Oct. 14, 1982, pp. 131–139.
Electronic Design News, Oct. 27, 1982, p. 165.

Primary Examiner—David Y. Eng
Attorney, Agent, or Firm—Edward D. Manzo

[57] **ABSTRACT**

A microcomputer comprises an integrated circuit device with processor and memory and communication links arranged to provide non-shared connections to similar links of other microcomputers. The communication links include message synchronisation and permit creation of networks or microcomputers with rapid communication between concurrent processes on the same or different microcomputers.

25 Claims, 15 Drawing Sheets



EXA

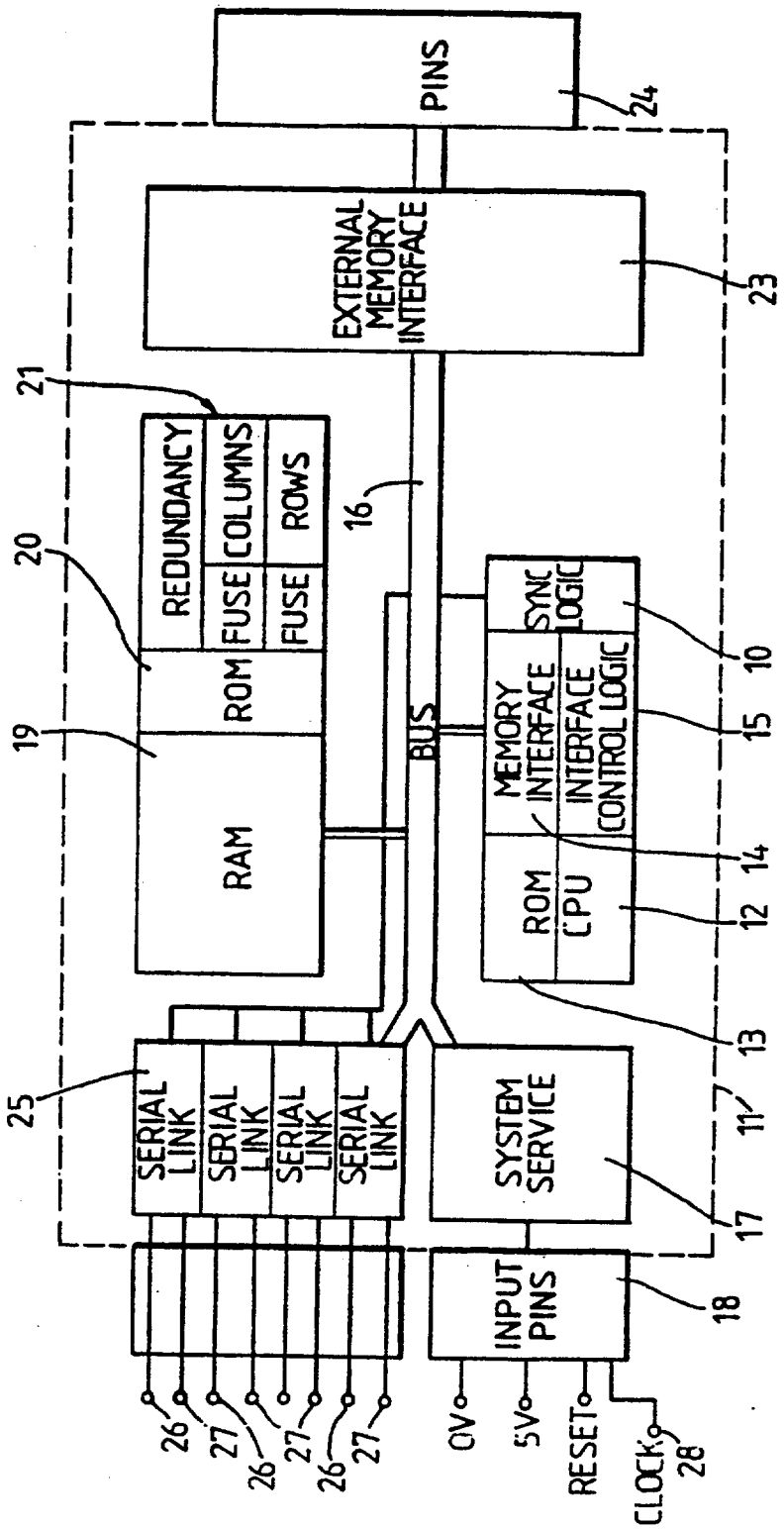

PINS
24
EXTERNAL MEMORY INTERFACE
23
21
REDUNDANCY
COLUMNS
ROWS
FUSE
FUSE
16
20
ROM
19
RAM
BUS
SYNC LOGIC
10
MEMORY INTERFACE CONTROL LOGIC
15
ROM
CPU
14
12
13
25
SERIAL LINK
SERIAL LINK
SERIAL LINK
SERIAL LINK
SYSTEM SERVICE
11
17
INPUT PINS
18
26
27
26
27
26
27
0V
5V
RESET
CLOCK
28

Fig. 1.

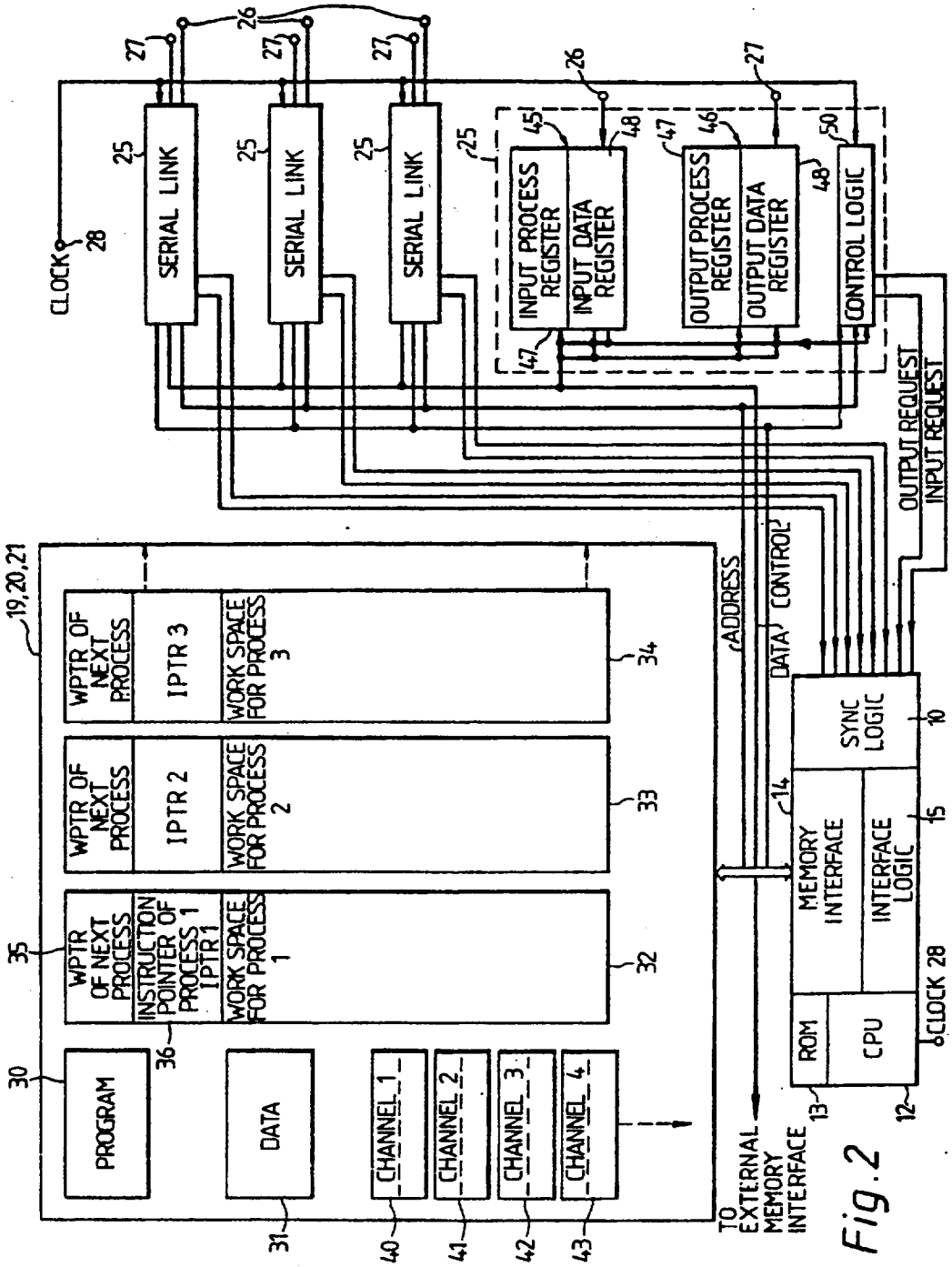

CLOCK
28
SERIAL LINK
SERIAL LINK
SERIAL LINK
INPUT PROCESS REGISTER
INPUT DATA REGISTER
OUTPUT PROCESS REGISTER
OUTPUT DATA REGISTER
CONTROL LOGIC
OUTPUT REQUEST
INPUT REQUEST
19,20,21
WPTR OF NEXT PROCESS
IPTR 3
WORK SPACE FOR PROCESS 3
WPTR OF NEXT PROCESS
IPTR 2
WORK SPACE FOR PROCESS 2
WPTR OF NEXT PROCESS
INSTRUCTION POINTER OF PROCESS 1 IPTR1
WORK SPACE FOR PROCESS 1
PROGRAM
DATA
CHANNEL 1
CHANNEL 2
CHANNEL 3
CHANNEL 4
ADDRESS
DATA
CONTROL
SYNC LOGIC
MEMORY INTERFACE
INTERFACE LOGIC
ROM
CPU
CLOCK 28
TO EXTERNAL MEMORY INTERFACE

Fig. 2

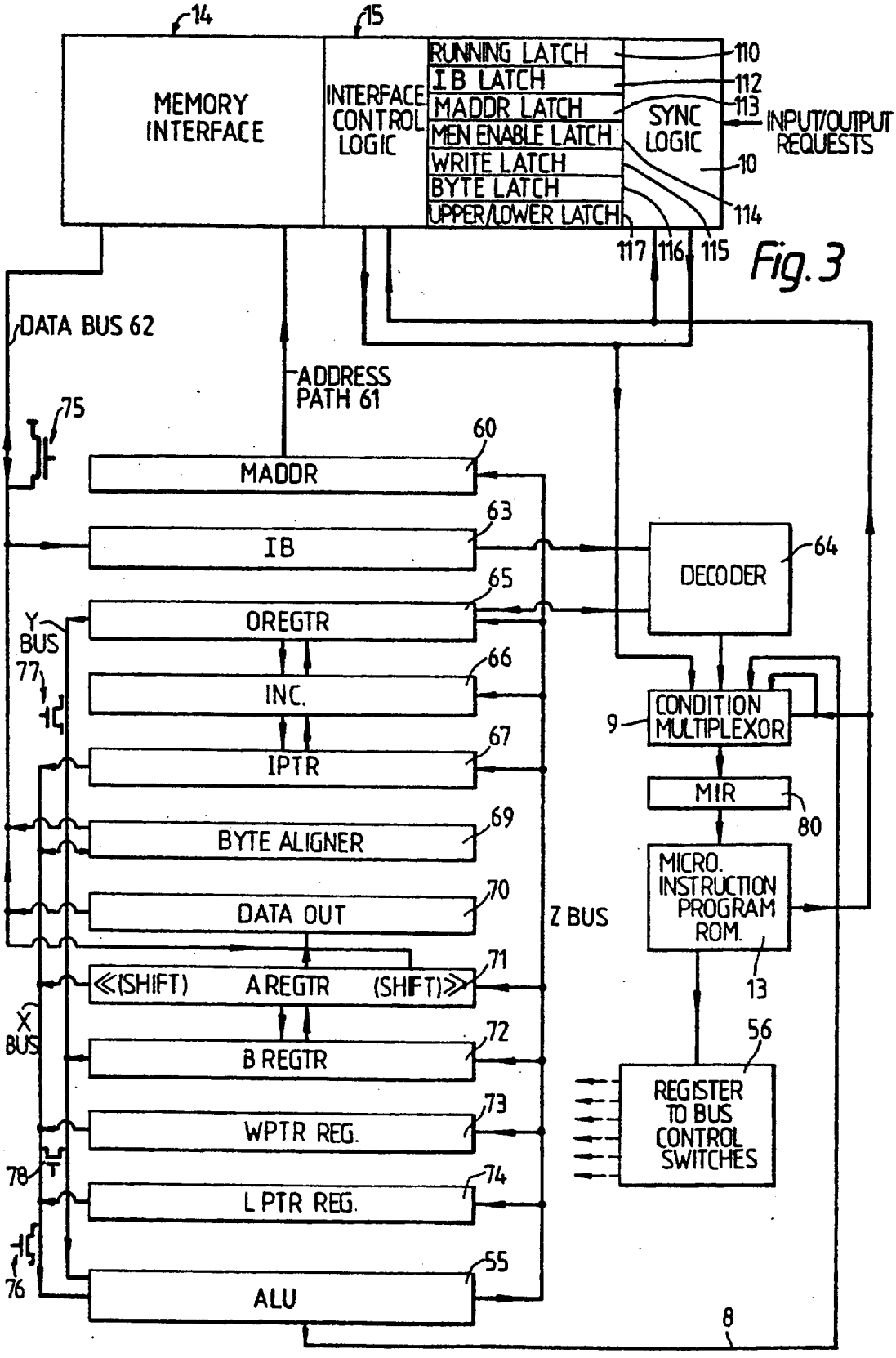

14
15
MEMORY INTERFACE
INTERFACE CONTROL LOGIC
RUNNING LATCH
IB LATCH
MADDR LATCH
MEN ENABLE LATCH
WRITE LATCH
BYTE LATCH
UPPER/LOWER LATCH
SYNC LOGIC
INPUT/OUTPUT REQUESTS
110
112
113
10
114
117
116
115
DATA BUS 62
ADDRESS PATH 61
75
60
MADDR
63
IB
64
DECODER
65
OREGTR
Y BUS
77
66
INC.
9
CONDITION MULTIPLEXOR
67
IPTR
MIR
80
69
BYTE ALIGNER
MICRO. INSTRUCTION PROGRAM ROM.
70
DATA OUT
Z BUS
71
≪(SHIFT) AREGTR (SHIFT)≫
13
X BUS
72
B REGTR
56
REGISTER TO BUS CONTROL SWITCHES
73
WPTR REG.
78
74
L PTR REG.
76
55
ALU
8

Fig. 3

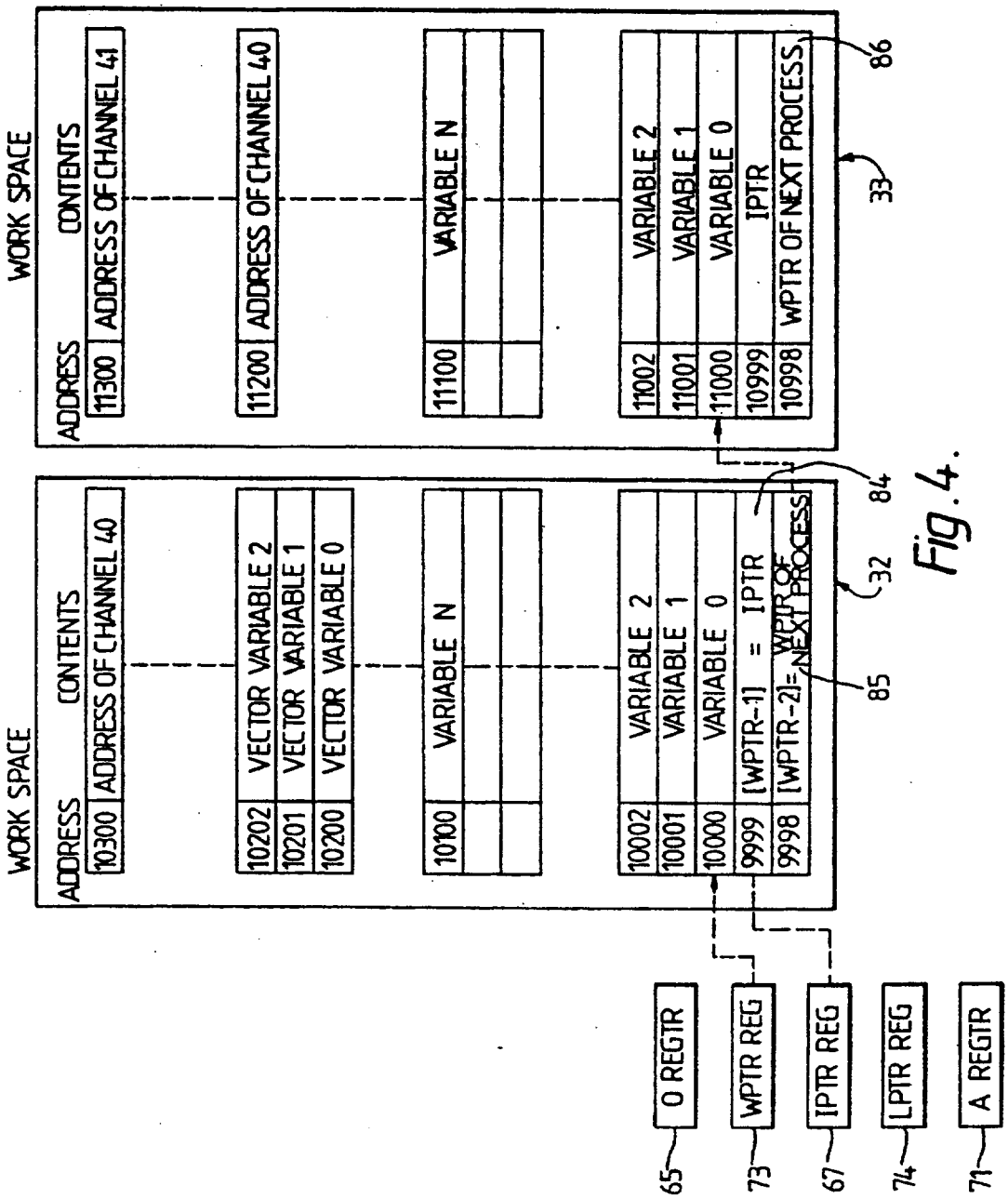

WORK SPACE
ADDRESS
CONTENTS
11300
ADDRESS OF CHANNEL 41
11200
ADDRESS OF CHANNEL 40
11100
VARIABLE N
11002
VARIABLE 2
11001
VARIABLE 1
11000
VARIABLE 0
10999
IPTR
10998
WPTR OF NEXT PROCESS
86
33
84
32
85
WORK SPACE
ADDRESS
CONTENTS
10300
ADDRESS OF CHANNEL 40
10202
VECTOR VARIABLE 2
10201
VECTOR VARIABLE 1
10200
VECTOR VARIABLE 0
10100
VARIABLE N
10002
VARIABLE 2
10001
VARIABLE 1
10000
VARIABLE 0
9999
[WPTR-1] = IPTR
9998
[WPTR-2]= WPTR OF NEXT PROCESS
0 REGTR
WPTR REG
IPTR REG
LPTR REG
A REGTR
65
73
67
74
71

Fig. 4.

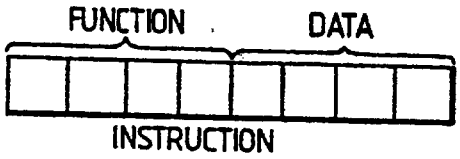


Fig.5.

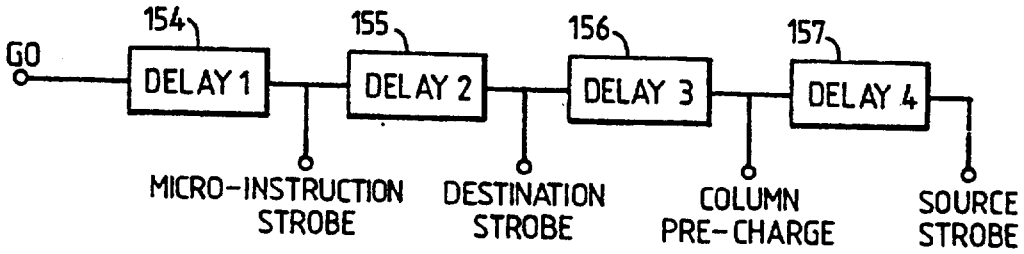


Fig.7

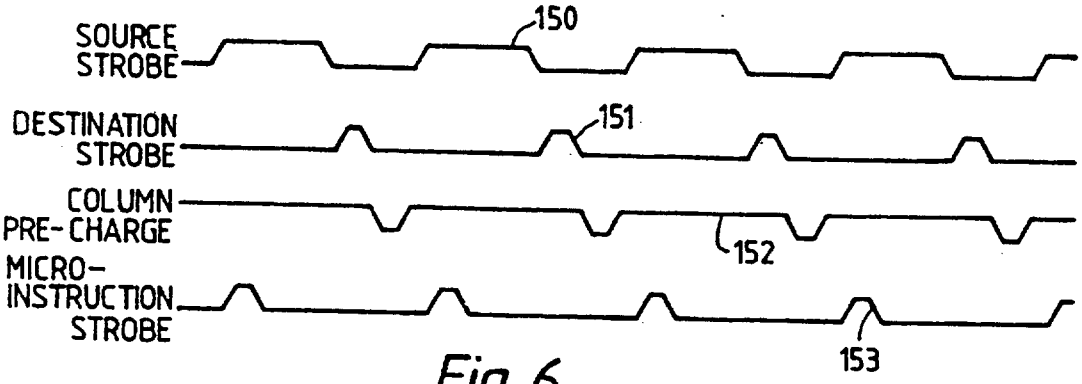


Fig.6

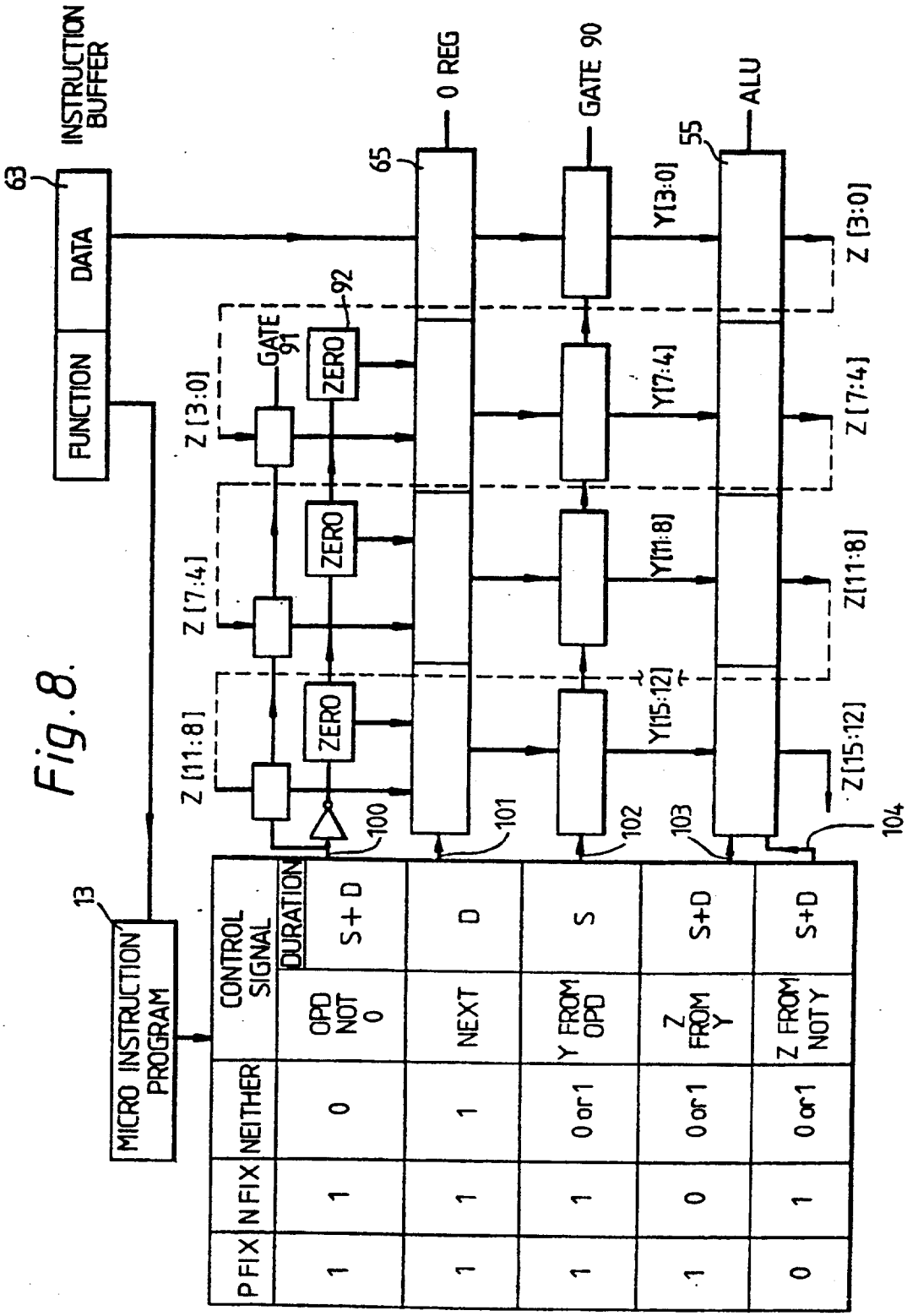

INSTRUCTION BUFFER
63
DATA
FUNCTION
13
MICRO INSTRUCTION PROGRAM
0 REG
65
GATE 90
55
ALU
Y[3:0]
Z [3:0]
GATE 91
92
ZERO
Z [3:0]
Y[7:4]
Z [7:4]
Z [7:4]
Y[11:8]
Z[11:8]
Z [11:8]
Y[15:12]
Z[15:12]
100
101
102
103
104
CONTROL SIGNAL
DURATION
S+D
D
S
S+D
S+D
OPD NOT 0
NEXT
Y FROM OPD
Z FROM Y
Z FROM NOT Y
NEITHER
0
1
0 or 1
0 or 1
0 or 1
N FIX
1
1
1
0
1
P FIX
1
1
1
1
0

Fig. 8.

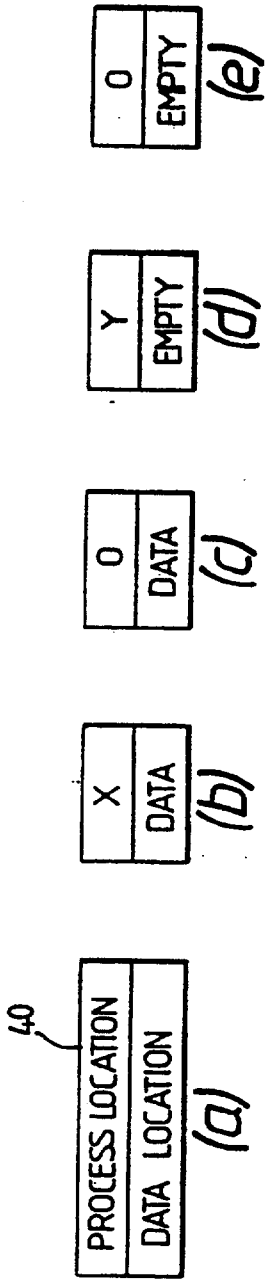


Fig. 9.

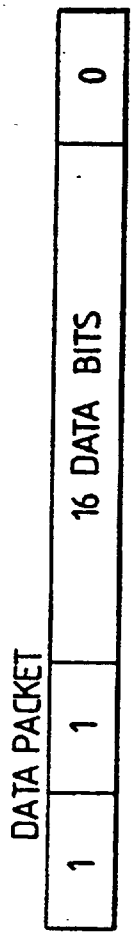


Fig. 13a.

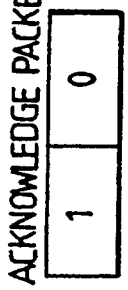


Fig. 13b.

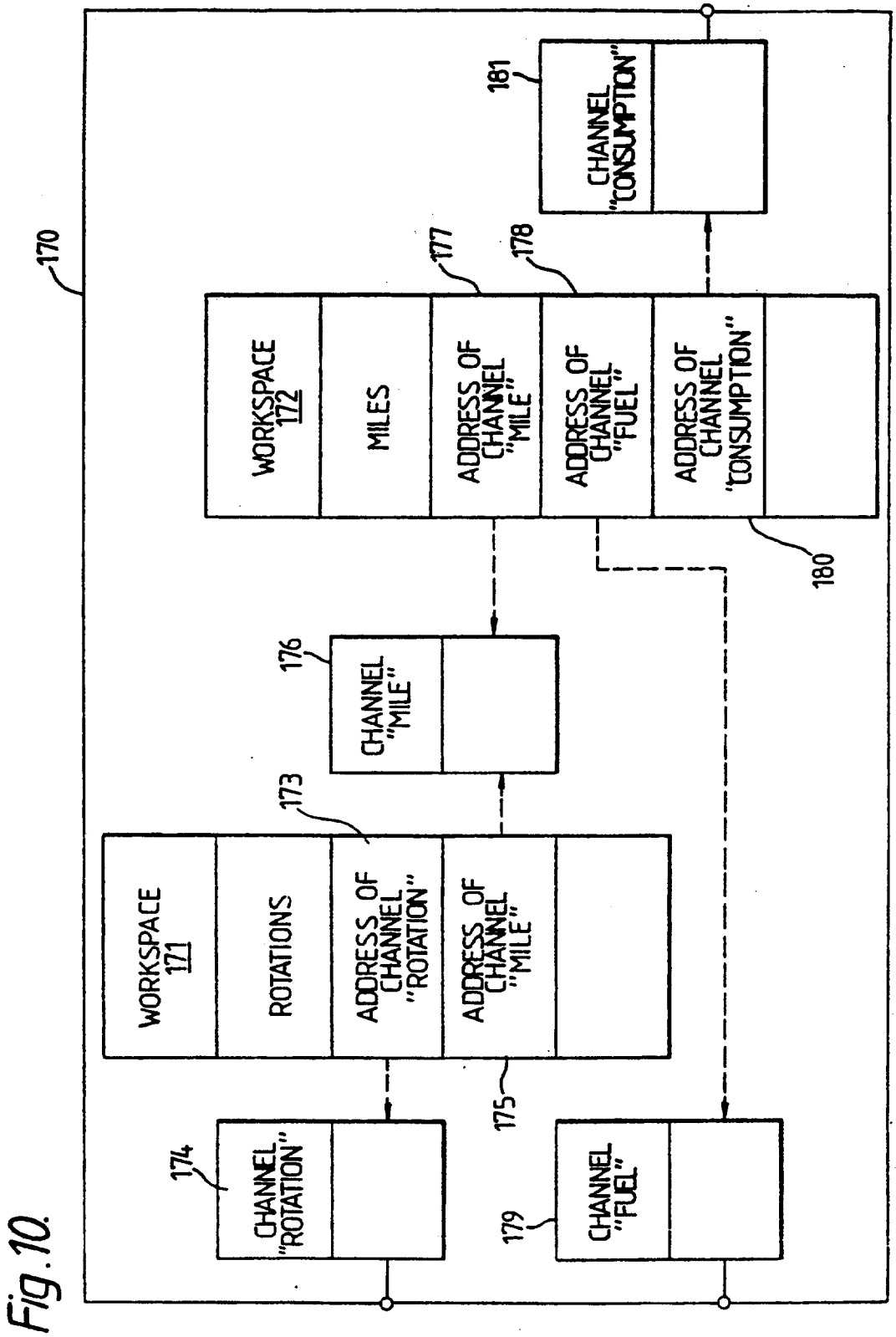

170
181
CHANNEL "CONSUMPTION"
177
178
WORKSPACE 172
MILES
ADDRESS OF CHANNEL "MILE"
ADDRESS OF CHANNEL "FUEL"
ADDRESS OF CHANNEL "CONSUMPTION"
180
176
CHANNEL "MILE"
173
WORKSPACE 171
ROTATIONS
ADDRESS OF CHANNEL "ROTATION"
ADDRESS OF CHANNEL "MILE"
175
174
CHANNEL "ROTATION"
179
CHANNEL "FUEL"

Fig. 10.

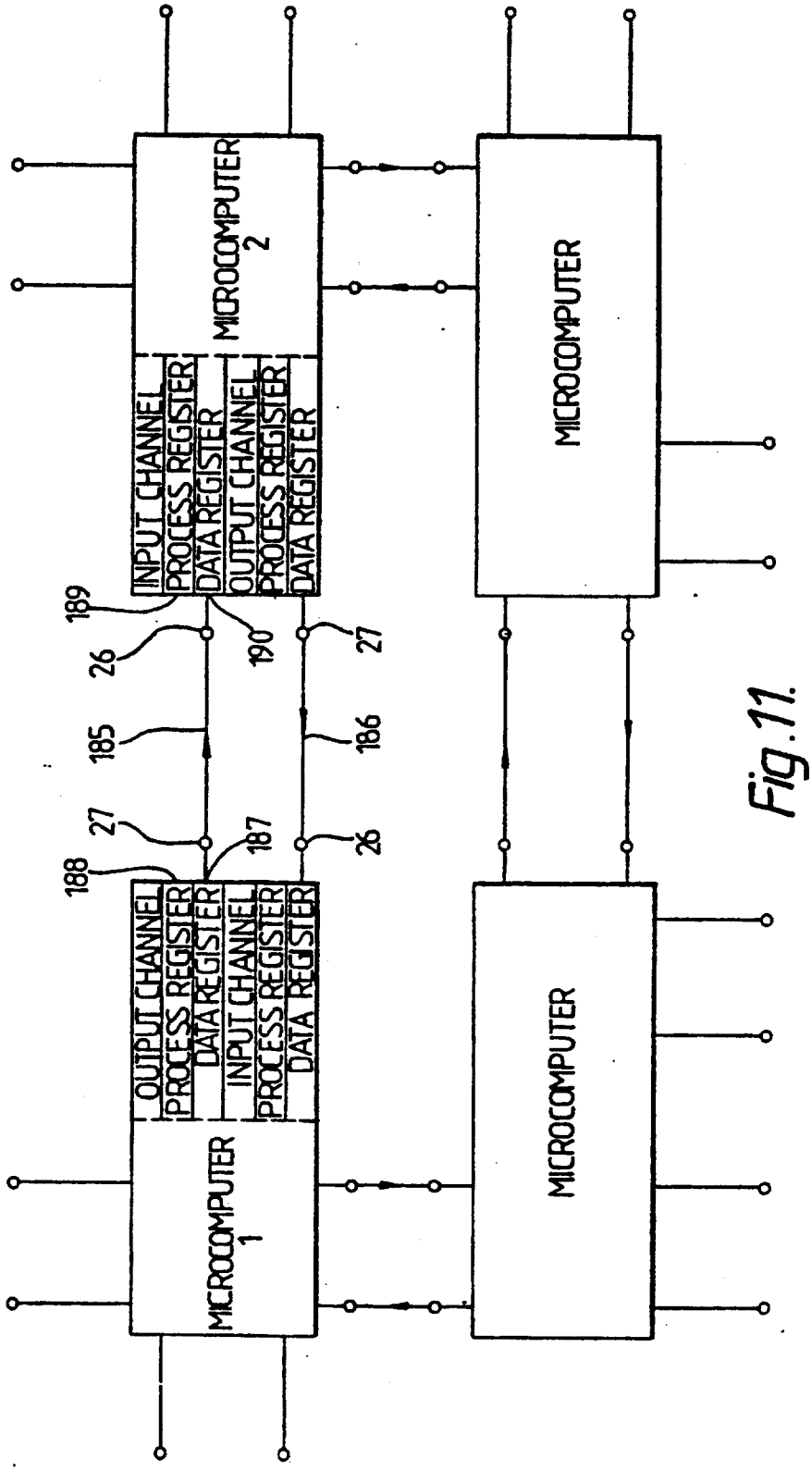

MICROCOMPUTER 2
INPUT CHANNEL
PROCESS REGISTER
DATA REGISTER
OUTPUT CHANNEL
PROCESS REGISTER
DATA REGISTER
MICROCOMPUTER
189
26
190
27
185
186
27
187
26
188
OUTPUT CHANNEL
PROCESS REGISTER
DATA REGISTER
INPUT CHANNEL
PROCESS REGISTER
DATA REGISTER
MICROCOMPUTER 1
MICROCOMPUTER

Fig. 11.

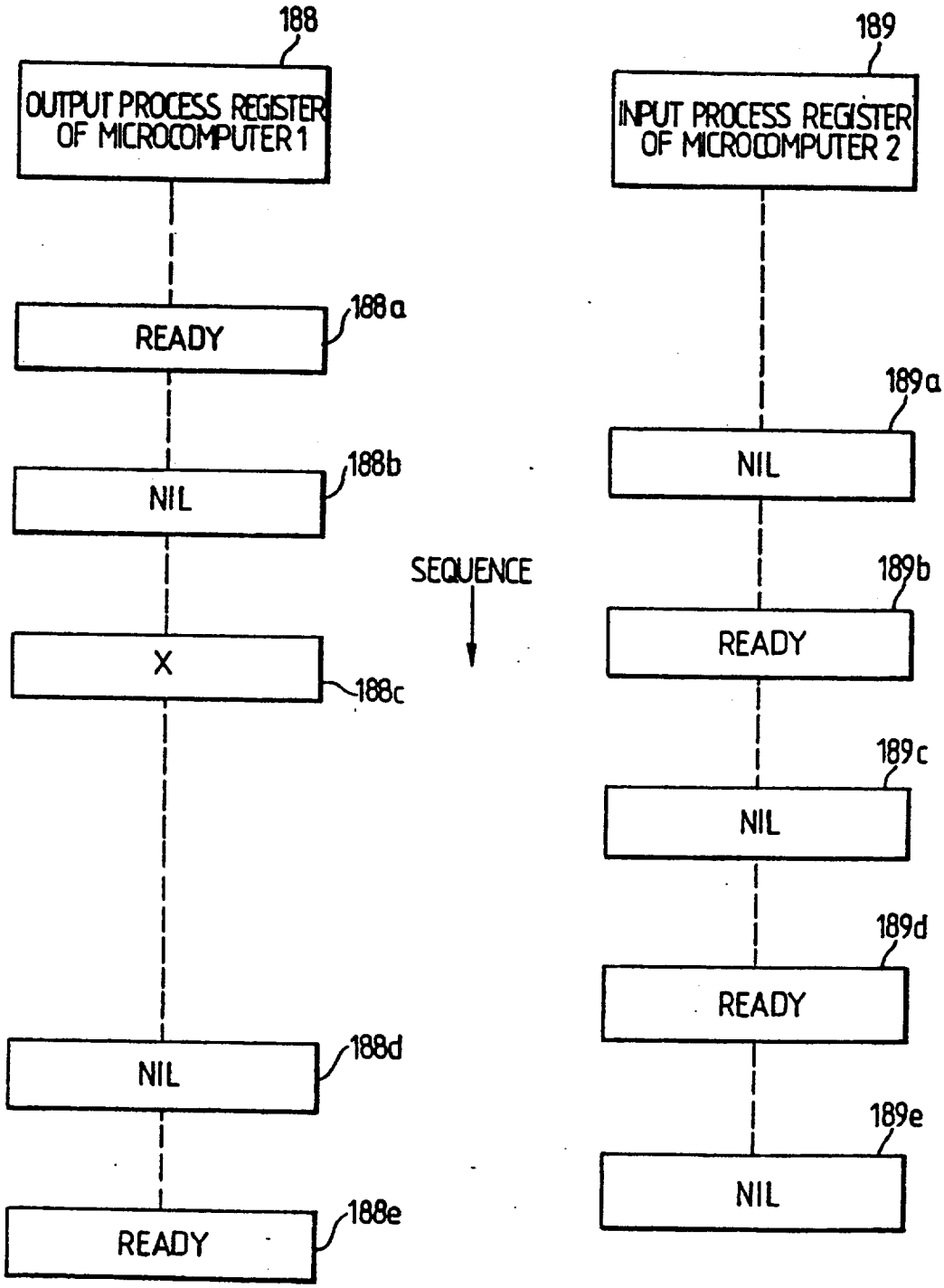
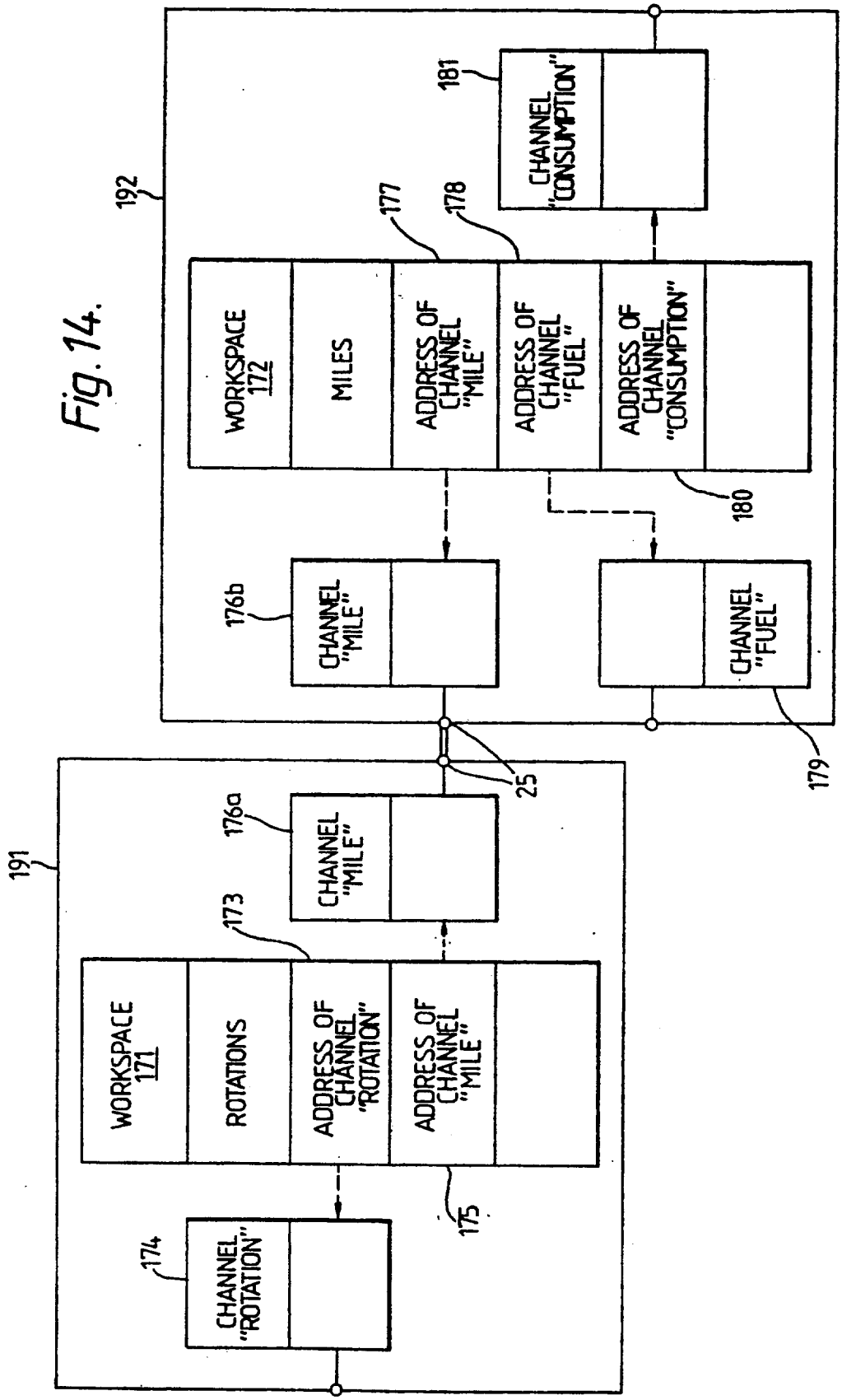

188
OUTPUT PROCESS REGISTER OF MICROCOMPUTER 1
READY
188a
NIL
188b
X
188c
NIL
188d
READY
188e
SEQUENCE
189
INPUT PROCESS REGISTER OF MICROCOMPUTER 2
189a
NIL
189b
READY
189c
NIL
189d
READY
189e
NIL

Fig.12.

Fig. 14.


192
181
CHANNEL "CONSUMPTION"
177
178
WORKSPACE 172
MILES
ADDRESS OF CHANNEL "MILE"
ADDRESS OF CHANNEL "FUEL"
ADDRESS OF CHANNEL "CONSUMPTION"
180
176b
CHANNEL "MILE"
CHANNEL "FUEL"
179
25
191
176a
CHANNEL "MILE"
173
WORKSPACE 171
ROTATIONS
ADDRESS OF CHANNEL "ROTATION"
ADDRESS OF CHANNEL "MILE"
175
174
CHANNEL "ROTATION"

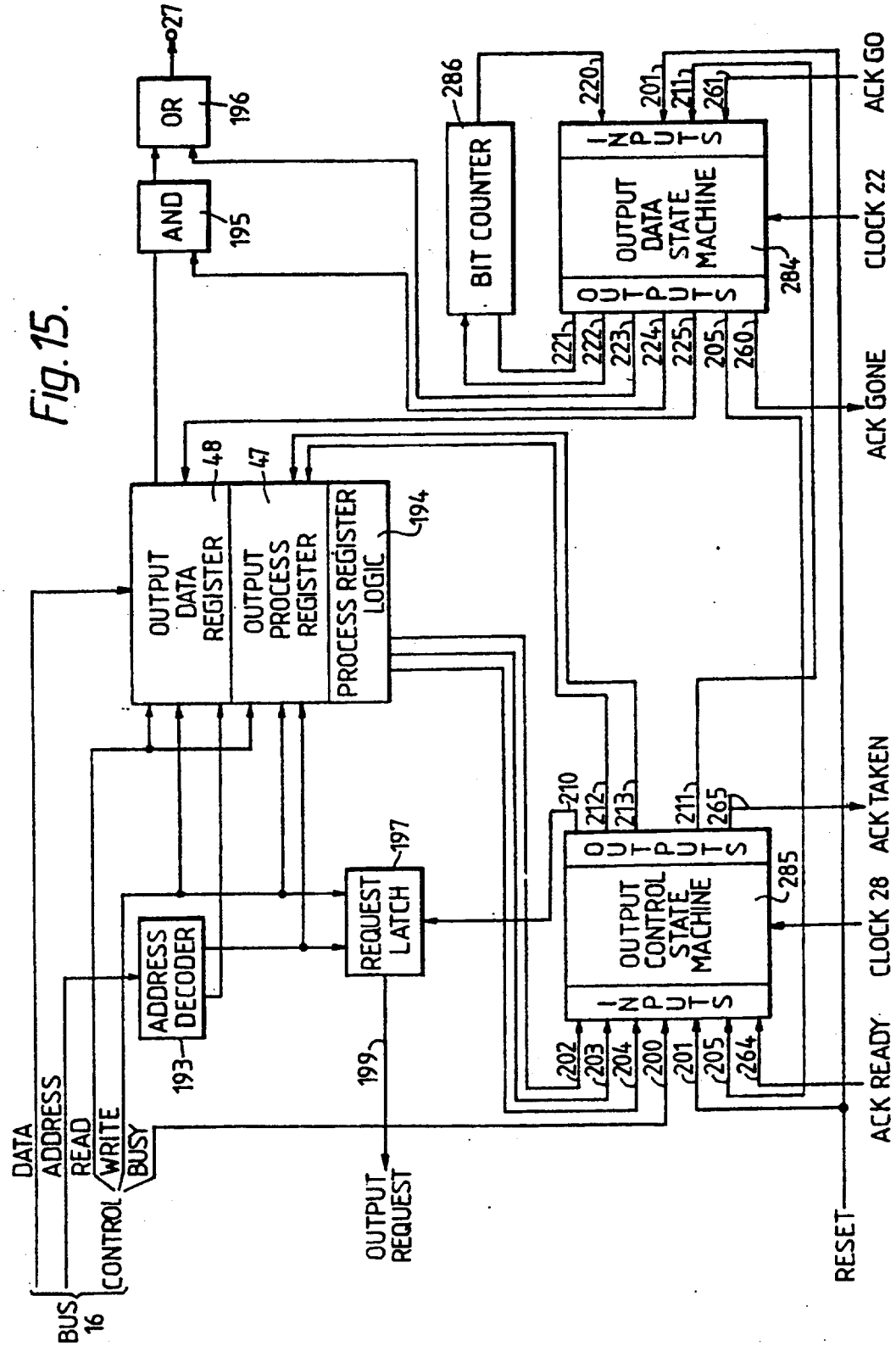
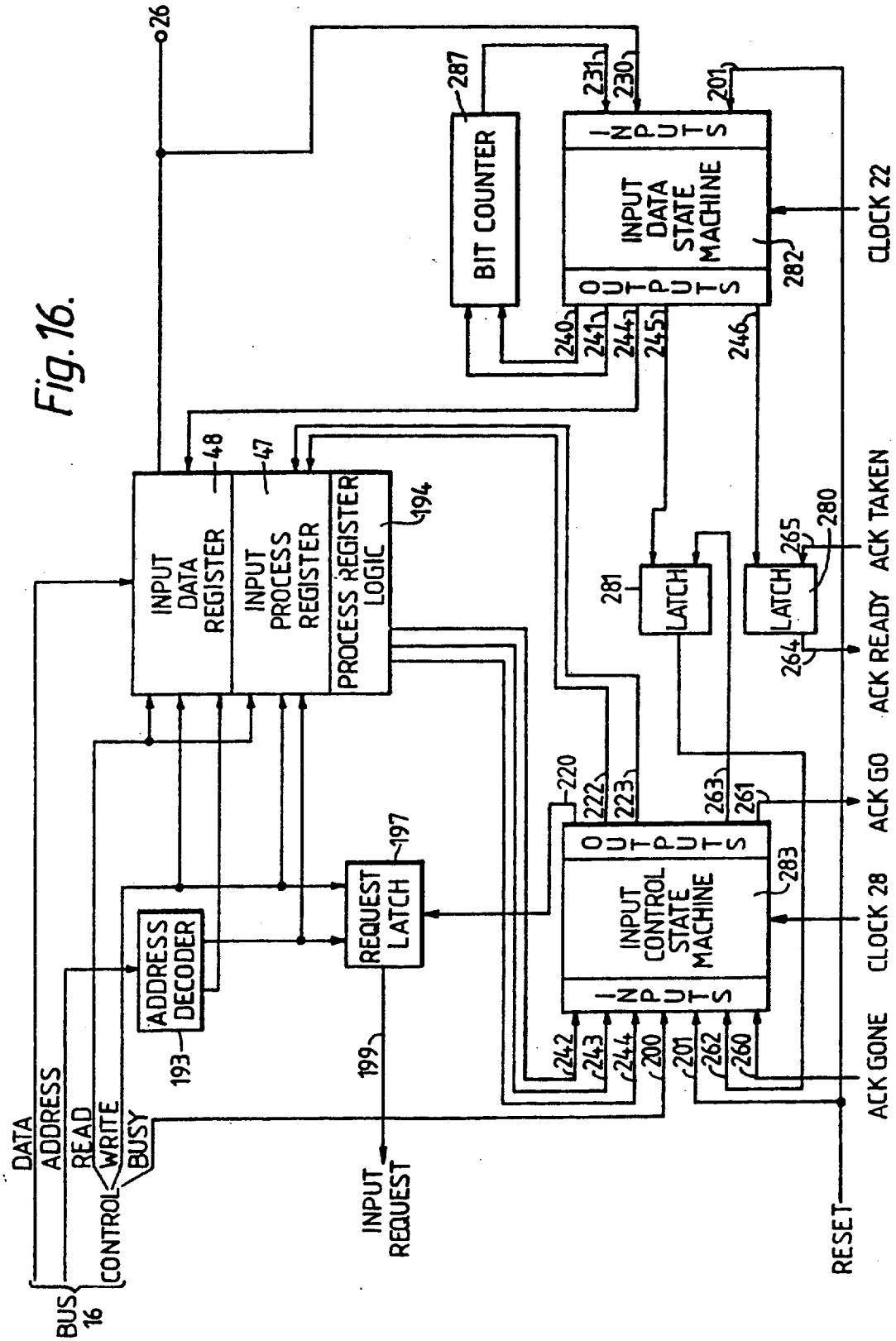

27
OR
196
AND
195
286
BIT COUNTER
220
201
211
261
INPUTS
OUTPUT DATA STATE MACHINE
OUTPUTS
284
ACK GO
CLOCK 22
ACK GONE
221
222
223
224
225
205
260
48
47
OUTPUT DATA REGISTER
OUTPUT PROCESS REGISTER
PROCESS REGISTER LOGIC
194
210
212
213
211
265
ACK TAKEN
197
REQUEST LATCH
OUTPUTS
OUTPUT CONTROL STATE MACHINE
285
CLOCK 28
INPUTS
ADDRESS DECODER
193
199
202
203
204
200
201
205
264
ACK READY
DATA
ADDRESS
READ
WRITE
BUSY
CONTROL
BUS 16
OUTPUT REQUEST
RESET

Fig. 15.

Fig. 16.


26
287
231
230
201
BIT COUNTER
INPUT DATA STATE MACHINE
INPUTS
OUTPUTS
CLOCK 22
282
240
241
244
245
246
48
47
194
INPUT DATA REGISTER
INPUT PROCESS REGISTER
PROCESS REGISTER LOGIC
281
LATCH
265
280
ACK TAKEN
264
ACK READY
220
222
223
263
261
ACK GO
197
REQUEST LATCH
ADDRESS DECODER
INPUT CONTROL STATE MACHINE
283
CLOCK 28
193
199
242
243
244
200
201
262
260
ACK GONE
DATA
ADDRESS
READ
WRITE
BUSY
CONTROL
BUS 16
INPUT REQUEST
RESET

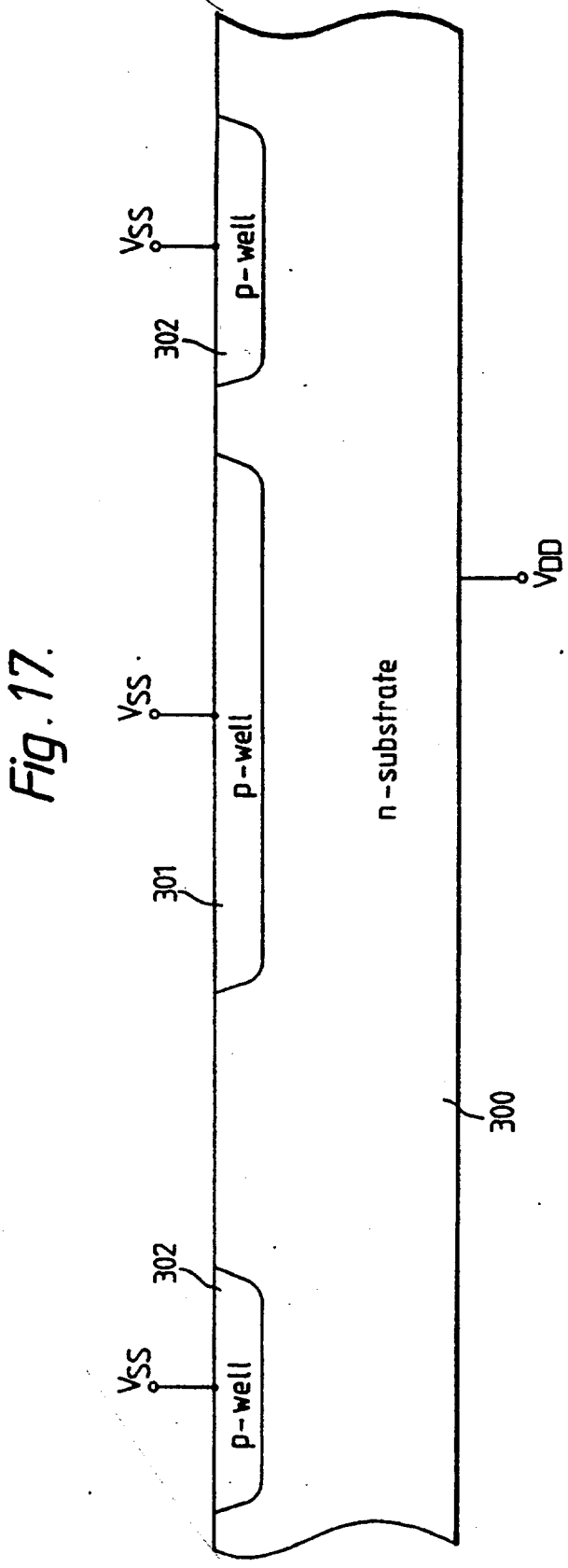


Fig. 17.

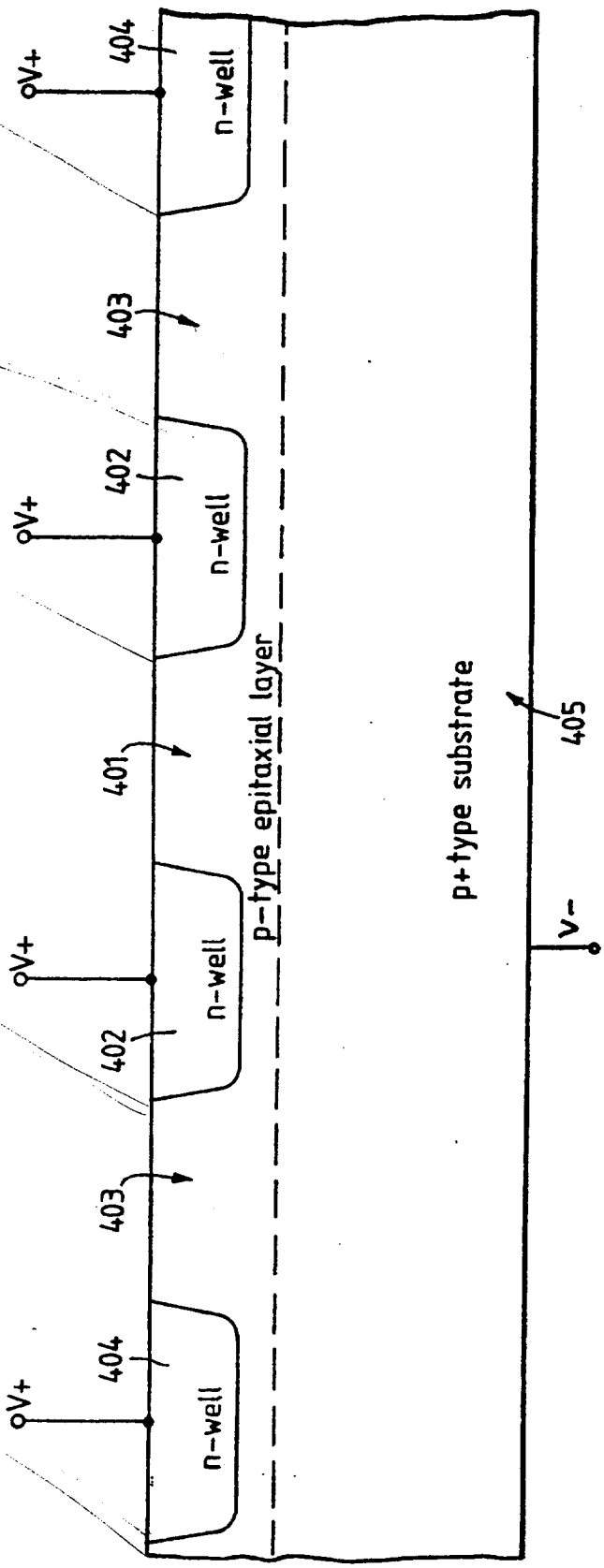

V+
404
n-well
403
V+
402
n-well
401
p-type epitaxial layer
p+type substrate
405
V+
n-well
402
403
V+
404
n-well
V-

Fig. 18.

MICROCOMPUTER WITH HIGH DENSITY RAM IN SEPARATE ISOLATION WELL ON SINGLE CHIP

This is a divisional application of co-pending application Ser. No. 938,380 filed on Dec. 9, 1986, now U.S. Pat. No. 4,967,326 which is, in turn, a continuation application of application Ser. No. 553,027, filed on Nov. 16, 1982, now abandoned.

The invention relates to microcomputers.

BACKGROUND OF THE INVENTION

Microcomputers generally comprise a processor and memory and may operate in accordance with a sequence of instructions derived form a stored program. The instructions may include a so-called "function" selected from a set of selectable functions and which define the operation which is carried out by the processor in response to that instruction. Processors may wish to communicate with external peripheral equipment including other microcomputers. For this reason; microcomputers are sometimes connected into a network with peripheral equipment or a plurality of microcomputers. This has generally been done through output ports or pins on the microcomputer and commonly these ports or pins have needed programming in order to communicate between processes on different microcomputers. Furthermore, it is conventional in microcomputers for communications, including the external communications, to occur through a bus which provides a bottleneck reducing the speed of operation of the microcomputer.

Consequently conventional microcomputers have not provided satisfactory building blocks for use in extended networks of microcomputers. Networks have generally caused loss of speed of operation and needed additional interface hardware.

OBJECTS OF THE PRESENT INVENTION

It is an object of the present invention to provide an improved microcomputer which is usable as a building block for a network of microcomputers.

It is a further object of the present invention to provide a network of interconnected microcomputers in which the network operates in the same manner as each individual microcomputer.

It is a further object of the invention to provide an improved microcomputer with a plurality of communication links which can operate concurrently.

It is a further object of the present invention to provide an improved microcomputer which may communicate with other microcomputers through links which avoid time delays normally encountered in shared buses.

It is a further object of the present invention to provide an improved microcomputer having a plurality of serial links thereby enabling it to be connected in communication with a larger number of other microcomputers.

It is a further object of the present invention to provide a microcomputer with independent serial links which allow direct connection with other microcomputers allowing operation of communication channels through the separate serial links concurrently.

SUMMARY OF THE PRESENT INVENTION

The present invention relates to different configurations of microcomputers on an integrated circuit chip with an on-chip memory using high density RAM that holds a sequence of instructions for execution by an on-chip processor. The RAM is protected from noise from on-chip transistors that operate independently of the RAM. The configurations re set forth in the claims, to which the attention is directed for a particular statement of the claimed subject matter. Generally, the configurations involve or relate to a substrate with first and second isolation regions, or first and second isolation wells, or a combination thereof.

It will be appreciated that the present invention relates to small size computers generally based on integrated circuit devices, but is not limited by how small the computer may be.

An example of a microcomputer in accordance with the present invention will now be described by way of example and with reference to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram snowing the main features of the microcomputer,

FIG. 2 is as more detailed block diagram of some of the components shown in FIG. 1 and in particular illustrates more fully the memory and serial links for external communication,

FIG. 3 shows further detail in block diagram form of part of the microcomputer and particularly illustrates the registers, data paths and arithmetic logic unit of the central processing unit as well as the interface between the central processing unit and other units of the microcomputer,

FIG. 4 illustrates the use of workspaces within the memory,

FIG. 5 illustrates schematically a form of instruction used in the microcomputer,

FIG. 6 shows in wave form the relative timing and duration of a plurality of timing control signals,

FIG. 7 illustrates the generation of timing control signals,

FIG. 8 illustrates the operation of the microcomputer of FIGS. 1 to 3 with variable length operands.

FIGS. *9a* to *9e* illustrate successive operations in one manner of communicating using a two word channel between two processes which are executed by the same microcomputer,

FIG. 10 illustrates the operation of two communicating processes on one microcomputer,

FIG. 11 shows a network of interconnected microcomputers, including detail of the serial link connection between two of them,

FIG. 12 illustrates a sequence of operations for effecting communication via serial links between two processes carried out on different microcomputers,

FIGS. 13*a* and 13*b* illustrate the format of data and acknowledge packets for transmission through serial links between two microcomputers,

FIG. 14 illustrates the operation of the same two communicating processes of FIG. 10 on two interconnected microcomputers,

FIG. 15 shows a logic diagram of one output serial link,

FIG. 16 shows a logic diagram of one input serial link,

FIG. 17 snows the chip formation which may be used for the microcomputer of FIG. 1, and

FIG. 18 shows an alternative chip formation which may be used for the microcomputer of FIG. 1.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The microcomputer described herein is an example of a Transputer (Trade Mark of Inmos International plc) microcomputer and comprises a single silicon chip having both a processor and memory as well as links to permit external communication. It is capable of carrying out a plurality of concurrent processes and effecting scheduling and communication between processes both on the same chip or separate chips. Each microcomputer has at least one K byte of memory in the form of programmable RAM on the same chip as the processor and the processor is capable of obeying programs in the chip's memory. The microcomputer has a plurality of communication links, herein called serial links, to enable it to be connected into a network of interconnected microcomputers so that any one microcomputer can be used as a building block for a network. The communication between any two microcomputers is effected by a serial link which provides one or more specific pin to pin connections each interconnecting two and only two microcomputers. Each link is not shared with other microcomputers or with any external memory. The microcomputer is provided with means for synchronization in data transmission between microcomputers within the network so that communication through a link between two microcomputers can be initiated by either the receiving or transmitting microcomputer.

The microcomputer contains a program with a plurality of sequential instructions each consisting of two parts, one part representing the function of the instruction and the other part representing data which is loaded into an operand register. In this way the function part of each instruction is of the same bit length regardless of the word length of the processor and in this way uniformity of function format and function bit length is achieved regardless of the word length of the processor. A further important feature of the microcomputer is that its operation is effected by use of a function set which is simple and efficient. The function set consists of a minimum number of functions. The function set includes direct functions which cause the processor to carry out an operation on the contents of the operand register. In a preferred arrangement it also includes one indirect function and two prefixing functions. The use of the indirect function allows a large number of processor operations to be used without increasing the number and size of data registers to perform the operations. Furthermore the use of a prefixing function provides for variable length operands.

By use of a microcomputer in accordance with this example, any required network of microcomputers can be formed by interlinking a number of microcomputers and the resulting network operates in the same way as any single microcomputer.

GENERAL DESCRIPTION OF THE STRUCTURE

The main elements of the microcomputer are illustrated in FIG. 1 on a single silicon chip 11 using p-well complementary MOS technology, which will be described in more detail with reference to FIG. 17. The components provided on the chip have been indicated in block form in FIG. 1 although it will be appreciated that the blocks are not intended to represent the relative size and positioning of the various components. On the chip there is provided a central processing unit (CPU) 12 which includes some read-only memory (ROM) 13. The CPU 12 is coupled to a memory interface 14 controlled by interface control logic 15. The CPU 12 incorporates an arithmetic logic unit (ALU), registers and data paths which will be described in more detail with reference to FIG. 3. The CPU 12 and memory interface 14 are connected to a bus 16 which provides interconnection between the elements on the chip 11. A service system 17 is provided with a plurality of input pins 18 including a zero volt supply, a 5 volt supply, a reset pin which may be activated to reset the microcomputer to a defined state, and a clock pin 28. The microcomputer is provided with a substantial amount of memory on the chip 11 and this is represented by a random-access memory RAM 19 and the ROM 20. The amount of memory on the chip should not be less than 1 K byte so as to provide sufficient memory capacity to allow the processor 12 to be operated without external memory. Preferably the memory on the chip is at least 4 K bytes. The division between RAM and ROM on the chip may be selected to suit the particular requirements for the microcomputer. The memory also includes redundancy 21 (this may be as described in our U.S. Pat. No. 4346459 or U.S. Pat. No. 4,389,715 or UK Patent Application No 8231055). This region 21 of memory has rows and columns selectively connectable by fuses as shown to replace defective regions of the memory 19 or 20 and thereby increase the production yield of chips which are satisfactory for use. The operation of the microcomputer includes timing control responsive to clock pulses from the pin 28. An external memory interface 23 is provided and connected to a plurality of pins 24 for connection to an optional external memory (not shown). In order to allow the microcomputer to be linked to other similar microcomputers to form a network, a plurality of serial links 25 are provided and in this example four are shown. Each serial link 25 has an input pin 26 and an output pin 27 each of which can be used to form a single pin to pin connection to corresponding output and input pins respectively of a further microcomputer. Each serial link is connected to a synchronization logic unit 10 comprising process scheduling logic which will be described in more detail below. Although the drawings show four serial links 25, three links, or even two links, may be used to form a single network but preferably at least six, and for example seven, such links are provided so that they may be fully interconnected in any desired array.

GENERAL DESCRIPTION OF USE OF CHIP MEMORY AND COMMUNICATION CHANNELS AND LINKS

FIG. 2 shows some of the elements of the microcomputer in more detail and in particular it illustrates the use of the memory on the chip. The microcomputer may be used to carry out a plurality of concurrent processes on the same chip and in FIG. 2 the operation of three concurrent processes have been shown. The memory is used to store the program 30 which may be stored in either ROM 20 or RAM 19. In this particular example the microcomputer is a 16 bit word device although it will be understood that other word lengths may be used. The program 30 consists of a sequence of instructions which in this example are each of 8 bit length and

this instruction length may remain the same even if the processor is of word length other than 16 bits. Each instruction is of the format shown in FIG. 5 where the most significant 4 bits represent the function of the instruction and the least significant 4 bits represent data. The program 30 incorporates no data other than that held in the designated part of each instruction. The manner in which the processor responds to each function and the way in which the data is handled depend on the particular function selected from a set of functions which will be described below, but the format of the function and data parts of each instruction is always the same. The memory also stores data 31 which may be stored in either the ROM 20 or RAM 19.

The microcomputer carries out a number of processes together, sharing its time between them. Processes which are carried out together are called concurrent processes. At any time, only one of the processes is actually being executed by the microcomputer and this process is called the current process. Each concurrent process to be effected by the microcomputer uses a region of memory called a workspace for holding the local variables and temporary values manipulated by the process. The address of the first local variable of each workspace is indicated by a workspace pointer (WPTR). Similarly for each concurrent process, an instruction pointer (IPTR) is used to indicate the next instruction to be executed from the sequence of instructions in the program relating to that particular process. In FIG. 2, which shows three concurrent processes, the workspace for process 1 is indicated by the numeral 32 and the corresponding workspaces for processes 2 and 3 have been marked 33 and 34. Each workspace consists of a plurality of addressable word locations and one word location 35 of each workspace is used to store the workspace pointer (WPTR) of the next process to be executed on a list of processes waiting to be executed. Thus, a linked list is formed in memory containing pointers to a sequence of workspaces for processes to be executed. If the processor is working on process 1 (see FIG. 2) and reaches a point where it is instructed that for the time being it is to stop executing that process, the CPU 12 will begin work on the next process, e.g. process 2. It will be directed to that next process by reading the workspace pointer in memory at location 35. In the preferred embodiment there is a known relationship between workspace pointer for any process and the address of the workspace pointer of the next process on the linked list, so that the next part of the linked list will be easily available from the current process workspace. For each process workspace, a further word location 36 stores the instruction pointer (IPTR) for that process. It will be appreciated that although workspaces for only three processes are shown in FIG. 2, the number may be varied depending on the number of concurrent processes to be carried out.

In order to allow communication between different processes carried out by the same microcomputer, a plurality of communication channels indicated by the numerals 40, 41, 42 and 43 are provided in the RAM section 19 of the memory. In this example each communication channel consists of two word locations in memory, one for use in identifying the process wishing to use the channel and the second for holding the data to be communicated through the channel. The operation of these channels will be described more fully with reference to FIGS. 9a-9e. FIG. 2 also shows in more detail the formation of one serial link 25. It is to be understood that each of the serial links is similarly formed. As indicated, the link 25 incorporates two channels 45 and 46 each forming a uni-directional communication channel. In this way the channel 45 is used as an input channel and the channel 46 as an output channel. Each channel consists of two registers each addressable in a manner similar to the two word locations of each of the channels 40 to 43. The two registers consist of a process register 47 used to indicate the process involved in the communication and a data register 48 for holding the data to be transmitted. The data register 48 in the input channel is connected to pin 26 and the data register 48 in the output channel is connected to pin 27. The operation of the two registers 47 and 48 is controlled by control logic 50 coupled to the synchronization unit 10. The operation of the serial links, control logic 50 and unit 10 will be described in more detail with reference to FIGS. 12 to 16.

The RAM section 19 of the memory is used to provide the workspaces 32 to 34 as well as the communication channels 40 to 43 and it may also be used for holding the program and data if required. The ROM 20 may be used for a variety of purposes such as for example holding an interpreter for a high level programming language or for storing "look-up" tables for standard operations. It may also be used to hold control programs for peripheral devices where the microcomputer is intended for a specific purpose.

CPU DATA PATHS AND REGISTERS

The central processing unit 12 and its operation will be more fully understood with reference to FIG. 3.

The CPU 12 includes an arithmetic logic unit (ALU) 55 and a plurality of data registers connected to three data buses, X bus, Y bus and Z bus. The operation of the registers and their interconnections with the buses is controlled by a plurality of switches diagrammatically represented by the reference numeral 56 and controlled by signals derived from a micro-instruction program contained in the ROM 13. It will be understood that these are switch means integrally formed in the chip construction. Communication between the CPU and the memory (19, 20, 21) is effected via a unidirectional address path 61 leading to the memory interface 14 and a bidirectional data bus 62 also connected to the interface 14. The registers, buses 61 and 62, and the X, Y and Z buses are connected as shown in FIG. 3. The registers are as follows:

Abbreviation	Register
MADDR	Memory address register 60 containing the address of the memory location required.
IB	Instruction buffer 63 for receiving sequentially from memory instructions of the form shown in FIG. 5.
OREGTR	An operand register 65 for receiving the data derived from an instruction in the instruction buffer 63.
IPTR REG	a register 67 which holds the instruction pointer (IPTR) of the current process.
DATA OUT	A register 70 for supplying data to the memory on the data bus 62.
AREGTR	A first (A) register 71 for holding an operand for the ALU 55.
BREGTR	A second (B) register 72 arranged as a stack with the AREG for holding operands for the ALU 55.
WPTR REG	A register 73 for holding the workspace pointer (WPTR) of the current process.
LPTR REG	A register 74 for holding a pointer to the workspace of the last process on the

-continued

Abbreviation	Register
	list of processes waiting to be executed.

As shown in FIG. 3, an incrementer 66 and a byte aligner 69 are also provided.

The data bus 62 is provided with a switch 75 operable to precharge the data bus line 62. The X and Y buses are respectively provided with similar switches 76 and 77 operable to precharge the X and Y buses. A further switch 78 is provided between the X and Y buses and is operable to cause signals on the two buses to merge.

The arithmetic logic unit 55 receives inputs from both the X and Y buses and is arranged to output to the Z bus. It provides a further output 8 to the micro-instruction program ROM 13, through a condition multiplexor 9, so as to control the operation of the data path in dependence on the output of the ALU 55.

The instruction buffer 63 is arranged to receive from the memory (19, 20, 21) via interface 14 and bus 62 a sequence of 8 bit words, herein called instructions, each of which has the format shown in FIG. 5 and consists of two parts. One part represents a "function" selected from the function set described below and the other part represents data. The instruction buffer 63 provides an output to a decoder 64 which separates the instruction into the function and data halves. The data half is loaded into the operand register 65 and the function half is decoded to provide an address to a micro-instruction register (MIR) 80. The identical procedure is followed for all instructions, regardless of function selected. Each instruction received by the instruction buffer 63 loads into the MIR 80 an address which causes the micro-instruction program in the ROM 13 to execute one or more micro-instructions controlling the switches 56 and interface control logic 15 so that at the end of each sequence of micro-instructions, an operation has been effected by the registers, control logic 15, and data paths of FIG. 3 corresponding to the selected function in the instruction. The operation of the micro-instruction program will be described more fully below.

All the registers shown in FIG. 3 apart from the instruction buffer 63 and the micro-instruction register 80 are 16 bit registers. It will be appreciated that in this example in which the processor is a 16 bit word processor, each 16 bit word location in the program contains two instructions, as each instruction is only 8 bits long. It is therefore necessary for the instruction pointer, which is held in the register 67 to be capable of pointing to a specific 8 bit byte in order to identify a single instruction from a program word location which incorporates two instructions. For this reason the program 30 (FIG. 2), in this example, is written into the bottom half only of the memory 19. In this example the memory has 64K words and consequently the program 30 is written into locations 0 to 32767 as the addresses of these locations can be represented by 15 bits only. This leaves an additional bit in the instruction pointer which can be used to identify which of the two bytes at each word address is necessary in order to identify a specific instruction. The micro-instruction ROM 13 contains 122 words, each of 68 bits. Each row of the ROM 13 contains 68 bits so that the ROM is arranged to provide 68 output signals at any time. The operation of the micro-instruction program will be described more fully below.

As can be seen from FIG. 3, the interface logic controller 15 is provided with a plurality of single bit state latches which are used to record the state of the memory interface. A latch 110, called a running latch, defines the source of instructions to be executed. If the latch 110 has state 1 the source of instructions is memory (this may be an external memory via the interface 23 if desired). If the latch has state 0, the source of instructions is one of the serial links 25 to allow instructions to be received from an external source. It may be necessary to go repeatedly to the same serial link 25 for two or more successive instructions whereas when the instructions are derived from memory, the instruction pointer IPTR is advanced for each instruction. An IB latch 112 records the state of the IB register 63. An MADDR latch 113 records the state of the MADDR register. A MEM ENABLE latch 114 records the state of the memory interface and has state 1 whenever the memory interface 14 is occupied. A WRITE latch 115 records that a write request has been made to the memory. The BYTE latch 116 records that a byte request has been made to the memory. An UPPER/LOWER latch 117 holds the least significant bit of byte addresses and is loaded from the least significant bit of the A register 71 when the content of the A register is shifted one place to the right.

FUNCTION SET

The function elements of the instructions which are received by the instruction buffer 63 are determined by the function set for the microcomputer. The function set is the list of available functions which can be selected when writing a program and to which the microcomputer is capable of responding.

There are three types of function in the function set.

Direct functions which use the contents of the operand register 65 as data (the contents of other registers may also be used as data).

Indirect function which use the contents of the operand register 65 to select one of a variety of "operations" using data in registers other than the operand register 65. The selectable "operations" are listed below the function set.

Prefixing functions which accumulate operands into the operand register 65.

The function set is as follows:

Code No	Abbreviation	Name
		FUNCTIONS
		Direct Functions
0	ldw	load from workspace
1	stw	store to workspace
2	ldpw	load pointer into workspace
3	ldwi	load from workspace and increment
4	ldv	load from vector
5	stv	store to vector
6	ldl	load literal
7	adl	add literal
8	j	jump
9	jnz	jump non zero
10	ldpc	load pointer into code
11	call	call procedure
		INDIRECT FUNCTIONS
13	opr	operate
		PREFIXING FUNCTIONS
14	pfix	prefix
15	nfix	negative prefix

The operations which may be effected by use of indirect functions are as follows:

OPERATIONS		
Code No.	Abbreviation	Name
0	rev	reverse
1	eqz	equal to zero
2	gt	greater
3	and	and
4	or	or
5	xor	exclusive or
6	add	add
7	sub	subtract
8	run	run process
9	pse	pause
10	join	join
11	sync	synchronize
12	ret	return
13	rot	rotate bytes
14	sr	shift right
15	sl	shift left

Prior to describing these functions and operations, the notion which is used herein will be set forth. The Transputer microcomputer is used preferably with OCCAM (Trade Mark of Inmos International plc) language, which is set forth more particularly in the booklet entitled *Programming Manual-OCCAM* published and distributed by Inmos Limited in 1983 in the United Kingdom, as well as Taylor and Wilson, "Process-Oriented Language Meets Demands of Distributed Processing", *Electronics* (Nov. 31, 1982), both of which are record in U.S. Pat. No. 4,704,678 and hereby incorporated herein by reference. OCCAM language is particularly well suited to concurrent processing. Because the preferred embodiment is particularly suitable for concurrent processing, the use of OCCAM language with the present example is quite appropriate. Other languages can be used with an appropriate compiler. In actual application, the programmer will write a program using OCCAM language and a compiler will convert this to particular instructions in customary fashion. Nevertheless, the functions and operations in the instructions are susceptible of description using OCCAM language to show what happens within the preferred embodiment of the microcomputer described herein. Thus, in describing these functions and operations, as well as examples of use, the following notation will be used:

NOTATION

1. PROCESS

A process starts, performs a number of actions, and then terminates. Each action may be an assignment, an input or an output. An assignment changes the value of a variable, an input receives a value from a channel, and an output sends a value to a channel.

At any time between its start and termination, a process may be ready to communicate on one or more of its channels. Each channel provides a one way connection between two concurrent processes; one of the processes may only output to the channel, and the other may only input from it.

An assignment is indicated by the symbol ":=". An assignment

v := e

sets the value of the variable v to the value of the expression e and then terminates. For example, x := 0 sets x to zero, and x := x 1 increases the value of x by 1.

An input is indicated by the symbol "?". An input

c? x

inputs a value from the channel "c", assigns it to the variable x and then terminates. An input

c ? ANY

inputs a value from the channel "c", and discards the value.

An output is indicated by the symbol "!". An output

c ! e

outputs the value of the expression e to the channel "c" and then terminates. An output

c ! ANY

outputs an arbitrary value to the channel "c".

The process SKIP terminates with no effect.

2. CONSTRUCT

A number of processes may be combined to form a sequential, parallel, conditional or alternative construct. A construct is itself a process, and may be used as a component of another construct. Each component process of a construct is written two spaces further from the left hand margin, to indicate that it is part of the construct.

A sequential construct is represented by

```
SEQ
  P1
  P2
  P3
  ...
```

The component processex P1, P2, P3 ... are executed one after another. Each component process starts after the previous one terminates and the construct terminates after the last component process terminates. For example

```
SEQ
  in ? x
  x := x + 1
  out ! x
```

inputs a value, adds one to it, and then outputs the result.

A parallel construct is represented by

```
PAR
  P1
  P2
  P3
  ...
```

The component processes P1, P2, P3 ... are executed called concurrent processes. The construct terminates after all of the component processes have terminated. For example,

```
PAR
  in ? x
  out ! y
```

allows an input to x and output from y to take place together.

Concurrent processes communicate using channels. When an input from a channel "c", and an output to the same channel "c" are executed together, communication takes place when both the input and the output are ready. The value is assigned from the outputting process to the inputting concurrent process, and execution of both concurrent processes then continues.

A conditional construct

```
IF
  condition 1
    P1
  condition 2
    P2
  condition 3
    P3
  ...
```

means that process P1 is executed if condition 1 is true, otherwise process P2 is executed if condition 2 is true, and so on. Only one of the processes is executed, and the construct then terminates. For example

```
IF
  x >= 0
    y := y+1
  x < 0
    SKIP
```

increases y only if the value of x is positive.

An alternative construct

```
ALT
  input 1
    P1
  input 2
    P2
  input 3
    P3
  ...
```

waits until one of input 1, input 2 ... is ready. If input 1 first becomes ready, input 1 is performed, and then process P1 is executed. Similarly, if input 2 first becomes ready, input 2 is performed, and then process P2 is executed. Only one of the inputs is performed, and then the corresponding process is executed and the construct terminates. For example:

```
ALT
  count ? ANY
    counter := counter + 1
  total ? ANY
    SEQ
      out ! counter
      counter := 0
```

either inputs a signal from the channel "count", and increases the variable "counter" by 1, or alternatively inputs from the channel "total", and outputs the current value of the variable "counter", and resets it to zero.

3. REPETITION

WHILE condition
  P

repeatedly executes the process P until the value of the condition is false. For example

WHILE x>5
  x := x−5

decreases x by 5 until its value is less than 5.

4. VARIABLES

A variable is either a simple variation, corresponding to a single word in store, or is one of a numbered set of variables called a vector. For example, v [3]:=0 sets the value of variable number 3 in the vector v to 0, and v[0]+1 increases the value of variable number 0 by 1.

A variable is introduced by a declaration such as

VAR v:
P

which introduces v for use in the process P.

5. PROCEDURES

A procedure definition allows a process to be given a name. For example

PROC square (n, sqr)
sqr:=n*n

defines the procedure "square".

The procedure name may be used as an abbreviation for the process. For example

square (x, sqrx)

means

sqrx:=x*x

6. EXPRESSIONS

An expression is constructed from operators, variables, numbers, the truth values TRUE and FALSE and the brackets (and). TRUE is a value consisting entirely of 1 bits, and FALSE is a value consisting entirely of 0 bits.

The operators +, −, *, / represent addition subtraction, multiplication and division as usual.

For the operators=, <>, > and <=, the result is produced as shown below:

x = y	true if x is equal to y
x <> y	true if x is not equal to y
x > y	true if x is greater than y
x <= y	true if x is less than or equal to y

For the operator, /\, \/ and ><, each bit of the result is produced from the corresponding bits of the operands according to the following table:

x	y	x /\ y	x \/ y	x >< y
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

For the NOT operator, each bit of the result is produced from the corresponding bit of the operand, according to the following table:

x	NOT x
0	1
1	0

For the operators << and >>

x<<y is the value of x moved y bits to the left, vacated bit positions being filled with 0 bits

x>>y the value of x moved y bits to the right vacated bit positions being filed with 0 bits

The above general OCCAM language notation will now be applied to the microcomputer of the example.

The register variables are defined as follows:

IPTR represents the contents of the instruction pointer register 67

WPTR represents the contents of the workspace pointer register 73

LPTR represents the contents of the list pointer register 74

AREG represents the contents of the A register 71

BREG represents the contents of the B register 72

OREG represents the contents of the operand register 65

A transfer from one register to another is represented by an assignment, e.g.:

BREG:=AREG

which means that the contents of the A register is copied to the B register, replacing the previous contents of the B register.

The memory in the transputer is represented by a vector:

memory

An individual word in memory is identified by subscripting the vector e.g.:

memory [AREG]

which means the contents of the word in memory whose address is the contents of the A register.

A transfer between memory and a register is similarly represented by an assignment e.g.:

memory [AREG]:=WPTR

which means that the contents of the word in memory whose address is the contents of the A register is replaced by the contents of the workspace pointer register.

Three procedures (PROC) "run", "wait" and "moveto" occur frequently in the following description. They are defined as follows, wherein link [process] represents the contents of the process register 47 of a serial link 25 and NIL represents a special value which is not the workspace pointer of any process. READY represents a further special value used by the serial links:

```
1   PROC run (w)
2     IF
3       w >< READY
4         SEQ
5           memory [ LPTR - 2] := w
6           LPTR := w
7       w = READY
8         SKIP
1   PROC wait
2     SEQ
3       memory [ WPTR - 1 ] := IPTR
4       for each external request from a serial link
5         SEQ
6           run ( link [ process ] )
7           link [ process ] := NIL
8       WPTR := memory [ WPTR - 2 ]
9       IPTR := memory [ WPTR - 1 ]
1   PROC moveto (w)
2     SEQ
3       IF
4         WPTR = LPTR
5           LPTR := w
6         WPTR <> LPTR
7           memory [ w - 2 ] := memory [ WPTR - 2 ]
8         WPTR := w
```

In the above procedures, line numbers have been added for reference purposes in explanation which will be given below.

Function and Operation Definitions

These are now set out below using the notation defined above.

load from workspace

Definition:
```
SEQ
  BREG := AREG
  AREG := memory [ WPTR + OREG ]
```
Purpose: to load the valve of a location in the current process workspace.

store to workspace

Definition:
```
SEQ
  memory [ WPTR + OREG ] := AREG
  AREG := BREG
```
Purpose: to store a value in a location in the current process workspace.

load pointer into workspace

Definition:
```
SEQ
  BREG := AREG
  AREG := WPTR + OREG
```
Purpose: to load a pointer to a location in the current process workspace
to load a pointer to the first location of a vector of locations in the current process workspace.

load from workspace and increment

Definition:
```
SEQ
  BREG := AREG
  AREG := memory [ WPTR + OREG ]
  memory [ WPTR + OREG ] := AREG + 1
```
Prupose: to load the value of a location in the current process workspace, and increment the location
to facilitate the use of workspace locations as loop counters, incrementing towards zero
to facilitate the use of workspace locations as incrementing pinters to vectors of words or bytes.

load from vector

Definition:
```
AREG := memory [ AREG + OREG ]
```
Purpose: to load a value from an outer workspace
to load a value from a vector of

-continued

Function and Operation Definitions	
	values to load a value, using a value as a pointer (indirection) - in this case OREG = 0
store to vector	
Definition:	SEQ memory [BREG + OREG] := AREG AREG := BREG
Purpose:	to store a value in a location in an outer workspace to store a value in a vector of values to store a value, using a value as a pointer (indirection) - in this case OREG = 0
load literal	
Definintion:	SEQ BREG := AREG AREG := OREG
Purpose:	to load a value
add literal	
Definition:	AREG := AREG + OREG
Purpose:	to add a value to load a pointer to a location in an outer workspace to load a pointer to a location in a vector of valves
jump	
Definition:	IPTR := IPTR + OREG
Purpose:	to transfer control forward or backwards, providing loops, exits from loops, continuation after conditional sections of program
jump non zero	
Definition:	IF AREG <> 0 IPTR := IPTR + OREG AREG = 0 SKIP
Purpose:	to transfer control forwards or backwards only if a non-zero value is loaded, providing conditional execution of sections of program and conditional loop exits to facilitate comparision of a value against a set of values
load pointer into code	
Definition:	SEQ BREG := AREG AREG := IPTR + OREG
Purpose:	to load into a A register the address of an instruction to load the address of a vector of data forming part of the program
call procedure	
Definition:	SEQ memory [WPTR - 1] := IPTR IPTR := AREG AREG := WPTR move to (WPTR + OREG)
Purpose:	to provide an efficient procedure call mechanism to facilitate code sharing, where two indentical procedures are executed on the same processor
Indirect Functions	
operate	
Definition:	operate (OREG)
Purpose:	perform an operation, using the contents of the operand register (OREG) as the code defining the operation required.
Prefixing Functions	
prefix	
Definition:	OREG := OREG << 4
Purpose:	to allow instruction operands which are not in the range 0–15 to be represented using one or more prefix instructions

-continued

Function and Operation Definitions	
negative prefix	
Definintion:	OREG := (NOT OREG) << 4
Purpose:	to allow nagative operands to be represented using a single negative prefix instruciton followed by zero or more prefix instructions.
Operations	
reverse	
Definition:	SEQ OREG := AREG AREG := BREG BREG := OREG
Purpose:	to exchange the contents of the A and B registers to reverse operands of asymmetric operators, where this cannot conveniently be done in a compiler
equal to zero	
Definition:	IF AREG = 0 AREG := TRUE AREG <> 0 AREG := FALSE
Purpose:	to test that A holds a non zero value to implement logical (but not bitwise) negation to implement A = 0 as eqz A <> 0 as eqz, eqz if A = 0... as jnz if A <> 0... as eqz, jnz
greater	
Definition:	IF BREG > AREG AREG := TRUE BREG <= AREG AREG := FALSE
Purpose:	to compare A and B (treating them as twos complement integers), loading −1 (true) if B is greater than A, 0 (false) otherwise to implement B < A by reversing operands to implement B <= A as (gt, eqz), and B >= A by reversing operands and (gt, eqz)
and	
Definition:	AREG := AREG / BREG
Purpose:	to load the bitwise AND of A and B, setting each bit to 1 if the corresponding bits in both A and B are set to 1, 0 otherwise to logically AND two truth values
or	
Definition:	AREG := BREG / AREG
Purpose:	to load the bitwise OR of A and B, setting each bit to 1 if either of the corresponding bits of A and B is set, 0 otherwise to logically OR two truth values
exclusive or	
Definition:	AREG := BREG >< AREG
Purpose:	to load the bitwise exclusive OR of A and B setting each bit to 1 if the correspoinding bits of A and B are different, 0 otherwise to implement bitwise not as (ldl −1, xor)
add	
Definition:	AREG := BREG + AREG
Purpose:	to load the sum of B and A to compute addresses of words or bytes in vectors
subtract	
Definition:	AREG := BREG - AREG
Purpose:	to subtract A from B, loading the result to implement A = B as sub, eqz

-continued

Function and Operation Definitions	
	A <>B as sub, eqz, eqz if A = B - as sub, jnz, . . . if A <> B as sub, eqz, jnz, . . .
run process	
Definition:	SEQ memory [AREG - 1]:= BREG run (AREG)
Purpose:	to add a process to the end of the active process list
pause	
Definition:	SEQ run (WPTR) wait ( )
Purpose:	to temporarily stop executing the current process to share the processor time between the processes currently on the active process list
join	
Definition:	IF memory [AREG] = 0 moveto (memory [AREG + 1]) memory [AREG] <> 0 SEQ memory [AREG] := memory [AREG] - 1 wait ( )
Purpose:	to join two parallel precesses; two words are used, one being a counter, the other a pointer to a workspace. When the count reaches 0, the workspace is changed
synchronize	
Definition:	IF memory [AREG] = NIL SEQ memory [AREG]:= WPTR wait ( ) memory [AREG] <> NIL SEQ run (memory [AREG]) memory [AREG]:= NIL
Purpose:	to allow two processes to synchronize and communicate using a channel
return	
Definition:	SEQ moveto (AREG) IPTR := memory [WPTR - 1] AREG := BREG
Purpose:	to return from a called procedure
rotate bytes	
Definition:	AREG := (AREG << 8) / (AREG << (bitsperword - 8))
Purpose:	to rotate the bytes in the a register to allow 8 bit byte values to be combined to form a single word value to allow a word value to be split into several component 8 bit values
shift right	
Difinition:	AREG := AREG >> 1
Purpose:	to shift the contents of the A register one place right
shift left	
Definition:	AREG := AREG << 1
Purpose:	to shift the contents of the A register one place left.

It will be seen that the above function set includes direct functions, indirect functions and prefixing functions. At the start of execution of any instruction, regardless of the function selected for that instruction, the predetermined set of bit positions in the instruction buffer **63** which receive the function part of the instruction are used to provide an input to the decoder **64** whereas the other predetermined bit positions in the instruction buffer **63** which represent the data part of each instruction are used to load the least significant four bit positions of the operand register **65**. If the function is a direct function, the processor then acts in accordance with the selected function on the contents of the operand register **65**. If the function is an indirect function, the contents of the operand register **65** are used to determine the nature of the operation to be carried out and the operation is effected on data held in other registers. At the end of any instruction in which the function is direct or indirect, the operand register **65** is cleared to zero. If the function is a prefix function, the processor operates to transfer existing data in the operand register

65 to positions of higher significance and then load into the vacated positions of lower significance data derived from the data part of the instruction.

At the start of each instruction, the instruction pointer is incremented. Consequently the instruction pointer always points to the next instruction to be executed.

The operand register 65 is used for several different purposes. The "data" which it receives with each instruction may be a literal value for use in a computation or in the case of an indirect function, it is the definition of the required operation. A further important use is that for some functions, the data value in the operand register 65 will be combined with the data in the workspace pointer register 73 to locate an address where the value of a particular variable is to be found or to be stored. For example, the workspace pointer register 73 will contain the workspace pointer WPTR of the current process. This points to a reference memory address for the workspace. Variables or other pointers will be defined and stored in that workspace at address locations which are offset by known amounts from the address pointed to by the workspace pointer WPTR. That offset will generally be specified by an instruction portion and stored in operand register 65. Indeed, the load and store from workspace instructions will implicitly refer to a memory location defined by the combination (illustratively the additive sum) of the contents of WPTR register 73 and the operand register 65. Furthermore, the contents of the operand register 65 will be combined with the contents of other registers such as the A register 71 or the IPTR register 67, for accessing vectors or for branching in the program. Examples of this will be given below.

It will be seen that the direct functions are selected to cover the most commonly required actions within the microcomputer in order to maximise efficiency of operation. By using 4 bits to represent the function element of each instruction, the function set uses codes 0 to 15 although no function has been allocated to code 12. Code 13 is used to indicate the indirect function which in this case is the "operate" function causing the least significant 4 bits of the instruction to be loaded into the operand register 65 in the usual way but the contents of that operand register are then used by the processor to determine an operation on data held in other registers. It will be appreciated that in this way the number of operations can be extended whilst maintaining uniformity of an 8 bit instruction. By use of the prefix or negative prefix functions before the "operate" instruction, the contents of the operand register 65 can be varied to provide a much greater selection of operations than is set out above. The use of pfix and nfix will be described in more detail below with reference to FIG. 8 but first it is necessary to describe further the operation of the micro-instruction program 13.

The micro-instruction program is the means of generating control signals which control the switches 56 and inferface control logic 15 (FIG. 3) in order to carry out the required "function" of each sequential instruction arriving in the instruction buffer 63 from the microcomputer program. The micro-instruction program consists of a list of micro-instructions stored in rows and columns in the ROM 13. The ROM 13 provides an output, called a micro-word, which may consist of 68 bits each providing a control signal and divided up into a plurality of different fields, each field consisting of a predetermined group of bit positions. The output at any one time is provided at selected bit positions depending on the micro-instruction selected. Each field may relate to a specific area of control, such as for example, one field controls which register is connected to the X bus, another field controls which register is connected to the Y bus, another field controls which register is connected to the Z bus, another field controls the action of the ALU 55 and another field controls feed back signals to the multiplexor 9 and MIR 80. One field controls the interface control logic 15 and provides micro-instruction output signals such as "Read", "Write" and "Next instruction required (NEXT)" to allow the microprogram to control communication between registers and the memory 19 through the interface 14.

The particular micro-instruction selected in the ROM 13 depends on the address in the MIR 80, which is a 7 bit register providing a row and column selection in the ROM 13. At the beginning of each instruction received by the instruction buffer 63 the "function" is decoded by the decoder 64 and is passed through the condition multiplexor 9 to provide an address for selection of the micro-instruction in the ROM 13. Some functions may require only one micro-instruction to carry out the function, in which case the ROM 13 provides a micro-word output dependent on the address decoded by the decoder 64 and the function is completed in one cycle of operation, herein called a minor cycle, of the ROM 13. Other functions require a succession of micro-instructions, and therefore minor cycles. In this case, the decoder 64 provides the MIR 80 with an address for the ROM 13 to select the first micro-instruction necessary for that function. Thereafter the micro program proceeds to execute a sequence of micro-instructions, each taking one minor cycle, and each micro instruction provides in a field of its output micro-word 7 bits for the MIR 80 so as to identify the address of the next micro instruction to be executed in the sequence. The least significant two bits of the MIR 80 may be conditionally set, so that the next minor instruction is selected as a result of conditions produced by a previous minor cycle, and fed back through the multiplexor 9 to effect the address in the MIR 80. This allows the next micro-instruction to be selected from four possible options depending on for example the values in the various registers shown in FIG. 3. If the two conditional bits of the MIR 80 are not set conditionally then the next micro-instruction indicated by the address in the MIR 80 is unconditionally executed. When all micro-instructions have been executed in order to achieve operation of the instruction in the instruction buffer 63, the control signal "NEXT" is generated in a field of the micro-word output of the ROM 13, thereby demanding the next instruction from the memory 19 to the instruction buffer 63.

Each minor cycle consists of two phases, a source phase and a destination phase. The control signals generated from the ROM 13 fall into three groups; those which are active only during the source phase, those which are active only during the destination phase and those which are active throughout the whole minor cycle. In order to control the occurrence and duration of the control signals, the timing control is arranged to provide four different strobe signals indicated in FIG. 6. These are a source strobe 150, a destination strobe 151, a column precharge strobe 152 and a micro-instruction strobe 153. The source strobe is a timing signal which allows a register to place its contents onto a bus and its duration is long enough to allow the arithmetic logic

unit to form a result. The destination strobe signals are arranged to allow registers to accept data from a bus. The micro-instruction strobe is used to generate the address of the next micro-instruction from the condition multiplexor 9. The column precharge strobe is used to precharge the bus lines X and Y to a high state ready for the next source strobe. The relative timing and duration of these strobes is shown in FIG. 6. They are generated by the arrangement shown in FIG. 7. The clock pulses from pin 28 (FIG. 1) generate a GO signal for the beginning of each minor cycle. This signal is passed through four successive delay units within the CPU 12 so that the micro-instruction strobe 153 is derived from the output of the first delay unit 154, the destination strobe 151 is derived from the output of the second delay unit 155, the column precharge signal 152 is derived from the output of the third delay unit 156 and the source strobe 150 is derived from the output of the fourth delay unit 157. The operation of the processor is therefore synchronised to the external clock input 28.

USE OF VARIABLE LENGTH OPERANDS

As already explained above, the microcomputer is capable of operating with a variable length operand. Although each instruction allocates 4 bit locations to an operand, it is possible to build up in the operand register 65 an operand up to 16 bits by use of the functions pfix and nfix corresponding to codes 14 and 15 in the function set set out above. This operation can best be understood with reference to FIG. 8. This indicates the operand register 65 having four sections each with 4 bits. The arithmetic logic unit 55 is indicated having four sections corresponding to 4 bits of increasing significance and the connection between the 0 register 65 and the arithmetic logic unit 55 is controlled via a gate 90 selectively controlling transmission through the Y bus to the arithmetic logic unit. The Y and Z buses are each shown separated into four parts, each handling four bits of data of different significance, e.g. Y[3:0] represents the part of the Y bus handling the four digits of least significance whereas Y[15:12] handles the four digits of greatest significance, and similarly for the Z bus. Each section of the operand register 65 other than the least significant 4 bits, can be supplied through a gate 91 from the Z bus or alternatively it can be fed with a zero from the gate 92. The instruction from the instruction buffer 63 in FIG. 8 is divided so that the least significant 4 bits are fed to the least significant 4 bit position of the 0 register 65 and thefunction element is used to select an address in the micro-instruction program 13 as previously described with reference to FIG. 3. The truth table of FIG. 8 indicates three alternative possibilities where the function corresponds to pfix or nfix or neither. It also lists the corresponding control signals which are fed onto lines 100 to 104 from the micro-word output of the ROM 13, and the duration of those signals.

The micro-word output control signals used in this case are as follows:

1. OPD NOT 0 - meaning that the operand register 65 is not supplied with zeroes if the truth table has a "1" but is supplied with zeroes if the truth table has a "0".
2. NEXT - meaning that the operand register 65 will be loaded with the next operand from the instruction buffer 63 if the truth table has a "1" but not if the truth table has a "0".
3. Y FROM OPD - meaning that the Y bus receives the operand from the operand register 65 if the truth table has a "1" but not if the truth table has a "0".
4. Z FROM Y - meaning that the Z bus output from the ALU 55 will receive data from the Y bus if the truth table has a "1", but not if the truth table has a "0".
5. Z FROM NOT Y - meaning that the ALU 55 will cause the signal on the Y bus to be inserted and passed to the Z bus if the truth table has a "1" but not if the truth table has an "0".

The duration of these five control signals in each minor cycle is indicated in FIG. 8 wherein S indicates duration in the source phase only, D indicates duration only in the destination phase and S+D indicates duration in both.

The micro-word control signal on line 100 operates the gates 91 and 92 to allow the Z bus to unload into the operand register 65 in response to the functions pfix and nfix whereas any other function causes the three most significant stages of the operand register 65 to be zeroed by an input through the gate 92. All instructions generate the control signal NEXT on the last minor cycle and this is applied to line 101 to cause the operand register 65 to be loaded with the next operand. Line 102 receives the signal "Y FROM OPD" and causes the operand register to be connected through the gate 90 to the Y bus for both pfix and nfix. Line 113 receives the control signal "Z FROM Y" and causes the arithmetic logic unit 55 to transmit to the Z bus the signal on the Y bus for pfix but not for nfix. Line 104 receives "Z FROM NOT Y" and allows the signal on Y to be inverted and supplied through the ALU 55 to the Z bus for nfix but not for pfix. The signals on lines 100, 103 and 104 exist throughout the source and destination phases of each minor cycle whereas the signal on line 101 exists only in the destination phase and the signal on line 102 exists only in the source phase. When the function is pfix, it can be seen that signals corresponding to a truth condition are supplied on lines 100, 101, 102 and 103 and in this way, the 4 bits of operand in the least significant section of the operand register 65 are advanced through the arithmetic logic unit to the next significant stage of the operand register 65 thereby allowing a further 4 bits of operand to be loaded into the least significant positions of the operand register 65. This operation is repeated each time an instruction is derived with pfix function up till a maximum of 16 bits of operand. Similarly if the function is nfix, the process is generally similar in allowing each successive 4 bits of operand to be moved up into a higher stage of the 0 register 65 without zeroes being written in after each instruction. This allows a negative operand to be built up to a maximum of 16 bits. The truth table indicates that if the function is neither pfix nor nfix, the control signal on line 100 causes zeroes to be fed into the three upper significant stages of the 0 register 65 (representing bits 15 to 4) at the end of that instruction.

SCHEDULING OF PROCESSES

As already indicated, the microcomputer may operate a number of concurrent processes. It therefore provides a system of scheduling to determine which process shall be performed at any particular time. At any one time the WPTR register 73 (FIGS. 3) holds the workspace pointer of the process currently being executed. However the workspace of the current process

and the workspaces of other processes waiting to be executed form a list in which one location of each workspace holds the workspace pointer of the next process on the list. Another location in each process workspace holds the instruction pointer identifying the next instruction which is to be carried out for that process when it becomes the current process. Furthermore the LPTR register 74 contains the address of the workspace for the last process currently waiting to be executed. In this way new processes can be added to the end of the list and the LPTR register 74 always indicates the current end of the list. The processor normally executes the processes on the list in sequence only advancing to a subsequent process when the current process executes a "pause" operation (code 9 in the operations list) or when the current process deschedules itself by executing a "join" operation (code 10 in the operations list) or a synchronize operation (code 11 in the operations list). In any of those situations, the current process ceases to carry out further instructions and the processor saves the instruction pointer IPTR in the process workspace as indicated at 36 in FIG. 2, and moves onto the next process which has been identified by the address of the next process, shown as 35 in FIG. 2 and then loads into the IPTR register 67 the IPTR for the new process. So that there is always at least one process running, a null process is provided and the null process is run when no other process is active.

The procedures "run", "wait", and "moveto" defined above are used in scheduling. A process will be "scheduled" when it is the current process or is on the linked list of processes which are waiting to be executed. A process becomes "descheduled" when it is taken off the linked list. A descheduled process will never be executed unless some other process or instruction schedules it, i.e. adds it to the end of the linked list. It will be recalled that LPTR register 74 (FIG. 3) is used to store the workspace pointer for the last process on the list. Hence, it must be adjusted whenever a process is added to the linked list. Also, when a process is to be scheduled, the CPU 12 must be able to determine which instruction is to be executed next for the process. This is done by storing in memory the appropriate instruction pointer IPTR, which is in IPTR register 67 while the process is current. Such storage is done, for example at memory location 36 (FIG. 2).

In describing these procedures, it will be convenient to refer to FIG. 4 which illustrates workspaces 32 and 33 more particularly, as well as registers 65, 67, 71, 73 and 74. FIG. 4 shows representative memory addresses and contents of the workspaces.

The process which has the workspace 32 is made the current process by inserting its workspace pointer WPTR into register 73. In this case, WPTR equals 10000. When the process becomes the current process the processor finds the next instruction to be executed by examining WPTR-1, i.e. the contents at memory location 9999, to find a pointer 84 to an instruction and loads this pointer in the IPTR register 67. While this is the current process, the processor will us the contents of IPTR register 67 to point to the next instruction.

During the processing, it will use variables whose addresses are formed by combining a reference value, such as the WPTR or the contents of the A register 71, and an operand placed in register 65. In a load from workspace operation an operand of "2" will refer to whatever is at memory location 10002 while the process corresponding to workspace 32 is current. When processing is to stop, the linked list is consulted. Elements 85 and 86 are part of the linked list. The processor will look at WPTR-2 to find WPTR 85 at memory location 9998, pointing to the next workspace. Pointer 85 will contain the number 11000 which points to workspace 33. If the process corresponding to workspace 33 is the last process on the linked list, then LPTR register 74 will contain the pointer 11000. No pointer 86 will be stored at memory location 10998 until some process is added to the linked list.

Turning now to the three procedures, PROC run (w) is used to schedule a process defined by w i.e., add it to the linked list. This procedure has been defined above and reference will now be made to that definition and the line numbers used in the definition.

If the value of w is the special value "READY" no action is performed. Further explanation of this will follow later with reference to communications between different microcomputers. Otherwise w is a pointer to a process workspace, and lines 5 and 6 will be executed in sequence. In line 5, LPTR means the contents of LPTR register 74, which is a pointer to the reference address for the workspace for the last process on the linked list. The memory whose address is LPTR-2 would contain the address of the workspace pointer for the next process, but as yet there is none because LPTR corresponds to the last process. Line 5 now assigns w (the workspace pointer in the process w) to memory location LPTR-2, so process w is now at the end of the linked list. At this point, the contents of LPTR register 74 points not to the last process w, but to the penultimate process. This is corrected in line 6 which enters into LPTR register 74 the workspace pointer for process w. Because of this step, further processes can be added to the linked list without deleting process w unintentionally, which would happen if LPTR register 74 were not updated. With reference to FIG. 4, if there are only two processes scheduled, as shown, and process w corresponds to a workspace whose pointer is 12000, PROC run (w) would enter 12000 in memory [10998] and enter 12000 into register 74.

The procedure called "wait" can be used alone or in combination with PROC run (w). By itself, it deschedules the current process and enables the system to execute the next process, executing it where appropriate in its program instead of at its first instruction. Reference will now be made to the previous definition of PROC wait. When procedure "wait" is called (line 1), a sequence is commenced (line 2) having four steps (lines 3, 4, 8 and 9). Lines 4–7 relate to external requests, and discussion of this can be deferred, although link [process] represents the contents of process register 47 of serial link 25 (FIG. 2). In line 3, memory [WPTR-1] is the memory space at the address WPTR-1, which is based on the reference address WPTR of the current process. That memory location is, in the preferred embodiment, used to point to the next instruction to be executed when the process is recommenced. The contents of IPTR register 67 always points to the instruction to be executed next for the current process. Hence, line 3 simply stores in memory (preferably on-chip) the pointer to the next instruction to be executed. If the procedure PROC run (w) has preceded PROC wait, then at this time, the current process (w) will have been added at the end of the linked list (by PROC run (w)1, LPTR register 74 will have been updated (also by PROC run (w)), and now the pointer to the next instruction for process w will have been stored at a

known location, memory [WPTR-1], with respect to the workspace pointer address (WPTR) for process w. Thus, process w is ready now to be deactivated. Line 8 looks to the linked list for the next process. Its workspace will be pointed to by the contents at address WPTR-2 of the current workspace. Hence, line 8 assigns to WPTR register 73 the workspace pointer for the next process on the linked list. Now the reference address has advance, and the system next finds out what the next instruction is for this process by looking at the pointer stored at the memory whose address is WPTR-1. To use FIG. 4, consider that workspace 32 is current and its process receives an instruction which includes PROC wait. Initially, WPTR is 10000. At line 8, register 73 is set to the contents found at memory address 9998, which will be the pointer 11000. At line 9, register 67 is set with the instruction pointer found at memory address 10999. Thus, if PROC run (w) is followed by PROC wait, the current process is added to the end of the list (its workspace pointer is stored on the linked list), the pointer to its next instruction is stored in memory, it is deactivated, and the next process on the linked list is commenced beginning at the proper instruction. All of this is done using only four registers. This arrangement permits the scheduling and descheduling of processes which are limited in number by only the amount of memory in the system.

The procedure named "moveto" can be used to set the workspace reference pointer to a different address in the workspace for the current process, without necessarily changing to a new IPTR. Thus, if a process has its reference workspace pointer at 10000, moveto (10200) could be used to set the registers to change the reference pointer to 102)) for this same process. This will be described as follows with reference to the previous definition of PROC moveto (w). Line 2 of the definition declares this is a sequence of steps. Lines 3 and 8 are equally offset from the left margin, so they both will be done in sequence. Assume that the system is not on the last process. Hence, line 4 will be false, so the system will jump to line 6. The condition at line 6 will be true, so line 7 will be executed. Line 7 sets the contents at memory address w-2 to the workspace pointer for the next process on the linked list. Next, line 8 changes the contents of the WPTR register 73 to the value w. Now register 73 points to a new reference address for the current process. At the customary offset (minus 2) from this new reference address will be found a pointer to the workspace for the next process to be scheduled. In the event that there is no next process, then line 4 will be true and LPTR register 74 will have its contents adjusted to point to w as the reference address for the last process (line 5), after which the register 73 for holding a pointer to the reference address of the current process will be adjusted to point to w.

Having now described FIG. 4 with reference to scheduling, some functions and operations will be further described with reference to FIG. 4.

load from workspace

The load from workspace function copies the contents at a specific memory location and puts it implicitly into the A register. This function and configuration of the preferred embodiment implicitly refers also to the memory whose address is defined by an offset from the current workspace pointer which serves as a reference. This reference address is always stored in the WPTR register 73, and the offset is contained in the operand register 65. The expression, "memory [WPTR . OREG]" therefore refers to the contents of the memory whose address is found by adding (summing) the contents of WPTR register 73 and register 65. A "load" refers to the A register 71, and the contents of the stack will be shifted down by one register, i.e. the contents of the A register will be shifted into the B register (to make room for the data to be loaded into AREG), and the contents of BREG will be loaded into the C register, if any. With reference to FIG. 4, if WPTR is 10000, then "load from workspace" using codes 0 2 will mean load variable 2 into the A register.

This "store to workspace" instruction implicitly means move whatever is in the A register 71 into the memory space whose address is offset from the reference address (contained in WPTR register 73) by the offset contained in the operand register 65. Also, the stack moves up (BREG moves into AREG, and CREG moves into BREG). Referring to FIG. 4 if WPTR=10000 and OREG=1, then this function means store the contents of the A register 71 into memory into memory location 10001, which is variable 1.

load pointer into workspace

The function "load pointer into workspace" does not store any data into the workspace. Instead, it loads the A register 71 with a pointer to a particular location in workspace. This will be used, for example, in connection with the "load from vector" instruction which references a particular portion of a vector which can be stored in the workspace. Thus, referring to FIG. 4 a workspace 32 will be referred to by the workspace pointer WPTR which is 10000. At a known location within the workspace, there can be a vector. The vector will have a plurality of locations such as 10200, 10201 and 10202. The beginning of the vector will be a particular offset (200) away from the workspace pointer (10000). Thus, to find the beginning of the vector, the offset (200) will be loaded into the operand register 65 and then the instruction "load pointer into workspace" will place the address 10200 into the A register, which points to the beginning of the vector. Thereafter, the "load from vector" operation will be used to find particular memory locations with respect to the beginning of the vector, and therefore it uses the offset in the operand register 65 but in combination with the A register 71 instead of the workspace pointer register 73.

load literal

The "load literal" instruction literally loads whatever is in the operand register 65 into the A register 71 (the top of the evaluation stack). With respect to FIG. 5, the last four bits of any given instruction will be loaded into the operand register 65, but by use of the prefixing functions, more than 4 bits can be stored in the operand register Illustratively, however, an instruction having the codes in decimal notation of 6 13 would mean load the number 13 into the A register 71 and shift the previous contents of the A register into the B register 72.

jump

The "jump" function is used for branching in a program. The instruction to be executed next by the processor for the current process is pointer to by the contents of the IPTR register 67 which contains the instruction pointer. The jump instruction adds the contents of the operand register 65 to the instruction pointer. Through use of the prefixing functions, the instruction

pointer can have values added to it or subtracted from it, to jump forward or backward in a program.

call procedure

The "call procedure" function uses the "moveto" procedure which was described above. "Call procedure" first stores IPTR in memory at the customary location for the instruction next to be executed (e.g. memory location 9999 in FIG. 4). Next it transfers into the instruction pointer register 67 the contents of the A register 71 which will have been loaded with the pointer to an instruction next to be executed after the "call procedure" function is completed. Then the A register 71 is loaded with the workspace pointer. Following this, the "moveto" procedute changes the reference pointer WPTR so that usually it points to a different address in the current workspace. It will be remembered that "moveto ( )" procedure will set the contents of the WPTR register 73 to whatever is within the parenthesis following the word "moveto". Thus, after a "call procedure," the system now has the workspace pointer pointing to a different location within the same workspace for the current process and is prepared to execute a different instruction which was previously contained in the A register 71. The converse operation is effected by use of the RETURN operation.

run process

This operation "run process" is generally used in the creation of a process which will have its own workspace and set of instructions. The A register 71 will have been loaded with a workspace pointer for the workspace for the new process, and the B register 72 will have been loaded with a suitable instruction pointer for the new process. Operation "run process" stores the instruction pointer in memory at the proper offset from the workspace pointer, and then it calls the procedure PROC run ( ), discussed above, using the workspace pointer in the parentheses. As discussed, this will schedule the new process, i.e. it will add the new process to the linked list.

pause

The "pause" operation appears in a program to prevent any single process from using the ALU 55 to the exclusion of other processes. This operation is inserted into loops by the compiler. This operation adds the current process to the end of the linked list, stores the necessary pointers, causes the current process to cease execution for the time being, and makes the next process on the linked list the current process. The contents of the evaluation stack are not preserved because "pause" is executed at a time when such contents can be discarded without harming the process.

join

This "join" operation is used for example when there are concurrent processes, and it is intended that they should all be at a point in the program at the same time. Consider an original process P(O) which at a certain point in the program, P(O) spreads into n concurrent subprocesses P(1), P(2), P(3) . . . P(n). When these are done, a final process P(n+1) is to execute. However, such final process should not occur until all of P(1) . . . P(n) have terminated. The "join" operation is used for this. A counter is set up in the workspace, and the A register 71 points to the memory where the count is stored. The count corresponds to the number of subprocesses (which are still active (not terminated). Each subprocess ends with a "join" operation. After a subprocess reaches its "join" operation, it checks the count. If the count is zero, then the program moves to the final process using the "moveto" procedure. If the count is not zero, the count is decremented by one count, and then the subprocess is caused to "wait" as described above. The other subprocesses are executed until zero count is reached.

synchronize

The "synchronize" operation is quite important to concurrent processing, for its use assures that two processes will be at the same point at some time. This will be discussed further in connection with FIG. 9 and the discussion entitled, "Communication Between Processes On The Same Microcomputer." Briefly however, if two processes X and Y on the same chip wish to communicate, presumably because one process is computing data which the other process needs, a channel 40, 41, 42 or 43 (FIG. 2) is used. Each process will have a "synchronize" operation. The first process to reach its "sync" operation will look at the channel. The channel address will have been loaded into the A register 71, so "memory [AREG]" refers to the channel. The expression "NIL" refers to a predetermined datum recognised as a nil. If NIL is found in a first part of the channel, by the first process, such process will place its workspace pointer into the first part of the channel and then will deschedule itself. It now waits for process Y to reach its "sync" operation. When this happens, process Y will check the first part of the same channel, and it will not find NIL but will instead find some workspace pointer. In response, it schedules process X (adds it to the end of the linked list). The first part of the channel returns to NIL. Generally there will be at least a second part to the channel where data for transfer from one process to the other will be placed. Also, synchronize operations generally occur in pairs. The first "sync" operations in two processes can cause the process to wait for data and then transfer it when it is ready. The second "sync" instructions cause acknowledgments. Thus, a process which is inputting data from a process will "sync". If the data is not ready, it will "wait". When the data is ready by the supply process, that supplying process will schedule the receiving process, which will then take the data. Then "sync" instructions by each acknowledge the transfer. The first "sync" by the process supplying the data will indicate that the data is ready to be taken.

COMMUNICATION BETWEEN PROCESSES ON THE SAME MICROCOMPUTER

As already explained, the microcomputer permits communication between processes which may be on the same microcomputer or on different microcomputers. For example, one process may be the measurement of distance travelled by a motor car and a second process the measurement of consumption of fuel relative to distance travelled for that vehicle. The first process may receive as an input, data representating rotations of the vehicle wheel and provide an output representing miles travelled. The second process may receive as an input data relating to fuel quantity consumed but it also needs to communicate with the first process to derive information about distance travelled before it can provide a useful output regarding fuel consumption relative to distance. In the case of process to process communications on the same microcomputer communication is

carried out in this example through the channels 40 to 43 indicated on FIG. 2. This operation involves the use of the synchronize operation, this requires a program instruction consisting of function code 13 and operation code 11 from the above list of functions and operations. Each channel 40 to 43 consists of two consecutive word locations in memory, one providing a "process location" and the other a "data location". The channel is a unidirectional communication channel which is shared by two and only two processes at any one time. When an active process x wishes to communicate with a process y on the same microcomputer, it follows a sequence which will be described with reference to FIGS. *9a* to *9e*. Firstly, process x identifies the address of the channel (marked 40) and loads the data it wishes to communicate into the data location of the channel. It also executes an instruction for a synchronize operation. Provided the process location does not already have the workspace pointer of the process y awaiting to receive the data, the synchronize operation causes the WPTR of process x to be recorded in the process location and uses a "wait" procedure to deschedule process x. This is the position shown in FIG. *9b*. Process x now waits until process y is ready to receive the data. When process y wishes to receive the data it carries out an instruction for a synchronize operation to see if the communication channel 40 is ready to transmit data. In carrying out this instruction, process y locates the workspace pointer of process x in the process location of channel 40 and as can be seen from the synchronize operation set out in the list of operations, the execution of a synchronize operation causes a "run" procedure to remove the workspace pointer of process x from channel 40 and add process x to the end of the list of processes waiting to be executed. This is the position in FIG. *9c*. Process y then reads the data from the data location of channel 40 and then operates a further instruction for a synchronize operation to indicate that it has received the data. This loads the workspace pointer of process y into the process location of channel 40 and causes process y to wait. This deschedules process y leaving the channel 40 in the condition shown in FIG. *9d*. Once the list on which process x is waiting reaches process x so that process x is reactivated, it performs a further instruction for a synchronize operation which now locates the workspace pointer of process y in the process location of channel 40 and this allows process x to continue to be operated. At the same time it causes a "run" procedure on process y so that process y is again added to the end of the waiting list of processes and is ready to run. The communication channel 40 is then in the condition shown in FIG. *9e* with process x continuing and process y waiting on the list. In this way, synchronization of communication is achieved by both processes operating a "handshake" operation in which both processes execute two instructions for synchronize operations one of which deschedules the process and that descheduled process is only put back onto the list when an appropriate signal has bee received from the other of the communicating processes.

A specific example of programs and instruction sequences necessary to carry out two communicating processes on the same microcomputer will now be described with reference to FIG. 10. This illustrates the two processes referred to above for measuring miles travelled and fuel consumption of a motor vehicle. The microcomputer 170 has in its memory space a first workspace 171 for the first process which is counting the variable "rotations" and a second workspace 172 for the second process which is counting the variable "miles". Workspace 171 has a word location 173 containing the address of the input channel 174 called "rotation" which forms part of a serial link arranged to receive a message for each wheel revolution from an external revolution detector (not shown). The workspace 171 has a further word location 175 containing the address of a two word memory channel 176 called channel "mile" which in this case receives an output from the process of workspace 171 indicating 1 mile of travel for each 1000 revolutions of the vehicle wheel.

For this first process the program using OCCAM language is as follows:

```
1. VAR rotations:
2. WHILE TRUE
3.   SEQ
4.     rotations:=0
5.     WHILE rotations <1000
6.       SEQ
7.         rotation ? ANY
8.         rotations:=rotations+1
9.     mile:ANY
```

Line numbers are not part of the program but have been added to facilitate explanation. Line 1 declares a variable to exist; it is called "rotations". Line 2 is an endless loop because the condition TRUE is always true. Start with zero rotations (line 4). Line 7 waits for any input from the channel names "rotation." When one is received, the variable "rotations" is incremented by one. Eventually there will have been 1000 rotations, and Line 5 will be false. Lines 6, 7 and 8 will then be skipped and Line 9 will output a datum to the channel named "mile".

The compiler will convert these OCCAM statements to the following machine instructions:

Instruction Sequence				Function code	Data	Program in OCCAM Language
						VAR rotations:
						WHILE TRUE
						SEQ
1.	L1:					
2.		ldl	0	6	0	rotations := 0
3.		stw	0	1	0	
4.	L2:					WHILE rotations <1000
						SEQ
5.		ldw	0	0	0	
6.		pfix		14	3	
7.		pfix		14	14	
8.		ldl	1000	6	8	
9.		opr	gt	13	2	
10.		jnz	L3	9	9	

-continued

	Instruction Sequence					
				Function code	Data	Program in OCCAM Language
11.		ldw	1	0	1	rotation ? ANY
12.		opr	sync	13	11	
13.		ldw	1	0	1	
14.		opr	sync	13	11	
15.		ldw	0	0	0	rotations:=rotations+1
16.		adl	1	7	1	
17.		stw	0	1	0	
18.		opr	pause	13	9	
19.		nfix		15	0	
20.		j	L2	8	0	
21.	L3:					
22.		ldw	2	0	2	mile ! ANY
23.		opr	sync	13	11	
24.		ldw	2	0	2	
25.		opr	sync	13	11	
26.		opr	pause	13	9	
27.		nfix	2	15	2	
28.		j	L1	8	7	

Once again, line numbers have been added for explanatory purposes only. Lines 1, 4 and 21 are simply reference locations in the program. Line 2 loads the value 0 into A register 71. Line 3 stores the data in the A register into workspace. Because the data part of the instruction is 0, there is no offset from the reference address for this process. Thus, the workspace pointer register 73 now contains a workspace pointer WPTR which points to a reference address in memory where 0 is stored for the variable "rotations". Line 5 loads the A register 71 from workspace. Because the data portion of the instruction (which would be loaded into operand register 65) is 0, the offset from the reference address WPTR of the workspace is 0. In lines 6, 7 and 8 the decimal value 1000 is to be added. This requires a prefixing operation because 1000 cannot be represented using four binary bits in the data portion of the instruction. Thus, function code 14 (pfix) is used. The decimal number 1000 in binary is 1111101000. Because this requires ten bits, and the data portion of standard instructions is four bits, three steps are required to load this value into the operand register. Line 8 includes the code for the "load literal" function, so at this time, the A register 71 will be loaded with the binary value of 1000. This causes the transfer of the contents of the A register (which are 0) to the B register 72.

Line 9 calls for an indirect function, the operation "greater than". This causes a comparison of the A and B registers. Unless the B register contents are greater than the A register contents, this operation will result in FALSE (0).

Line 10 is the "jump nonzero" operation. If the results of line 9's operation were true, then the A register would be set to a nonzero value, and line 10 would cause a jump of 9 lines forward, indicated by the number "9" in the data part of the code. This should jump the program ahead to line 21, the output portion. Assuming that 1000 rotations have not yet been counted, line 11 is next executed. This load from workspace function has an operand of +1, which means the offset from the reference address is +1. At this be found the address of the channel named "rotation" and this address will be loaded from the workspace into the A register 71. Line 12 causes a synchronize operation. Line 13 again loads the address of the channel "rotation" and line 14 again synchronizes to complete the input operation. In this simple example, no data is transferred. Line 15 now loads the variable which is in workspace, offset 0, into the A register, i.e. loads the current value of "rotations" into the A register. Line 16 literally adds the data value 1 to the contents of the A register. Line 17 stores the contents of the A register in the workspace at an offset equal to 0. Hence, the variable "rotations" has now been incremented in response to receipt of data from the channel "rotation". Line 18 is a pause operation which allows the next process to be executed, adding this present process to the end of the list. Note that at this point in the program, the contents of the A register 71 and B register 72 are not relevant to the process. Lines 19 and 20 executes a jump backwards using the negative prefix function. Line 22 loads the contents of the workspace which is offset 2 locations from the reference location. This will be the address of the channel named "mile" and it will be loaded into the A register. A sync operation is performed at line 23. The output is completed by a further "sync" which occurs at line 25. At line 26 another pause is inserted to cause the next process to schedule and to add this process to the end of the linked list. At lines 27 and 28, a jump backwards is executed using negative prefixing. The second process relating to the variable "miles" uses the workspace 172. The workspace 172 has a word location 177 containing the address for the "mile" channel 176 which is used to provide an input to the process of workspace 172. A further word location 178 has the address of a second input which in this case is a channel 179 called channel "fuel" forming part of a serial link arranged to receive a message from an external fuel gauge (not shown) each time a gallon of fuel is consumed. The workspace 172 has a further word location 180 having the address of an output channel 181 called channel "consumption" forming part of a serial link arranged to output the distance travelled while the last gallon of fuel was consumed. Clearly the process in workspace 172 needs to communicate with the process in workspace 171 in order to obtain via channel 176 messages indicating the number of miles travelled. The instruction sequence and program for the process in workspace 172 are as follows:

Instruction sequence		
Function code	Data	Program in above defined OCCAM
		VAR miles:

-continued

Instruction sequence			Function code	Data	Program in above defined OCCAM
					SEQ
L1:					
	ldl	0	6	0	miles := 0
	stw	0	1	0	WHILE TRUE
L2:					ALT
	ldw	1	0	1	mile ? ANY
	ldv	0	4	0	
	opr	eqz	13	1	
	jnz	13	9	9	
	ldw	1	0	1	
	opr	sync	13	11	
	ldw	1	0	1	
	opr	sync	13	11	
	ldw	0	0	0	miles := miles + 1
	adl	1	7	1	
	stw	0	1	0	
	pfix	1	14	1	
	j	L4	8	0	
L3:					
	ldw	2	0	2	fuel * ANY
	ldv	0	4	0	
	opr	eqv	13	1	
	jnz	L4	9	12	
	ldw	2	0	2	
	opr	sync	13	11	
					SEQ
	ldw	3	0	3	consumption ! miles
	ldw	0	0	0	
	stv	1	5	1	
	opr	sync	13	11	
	ldw	3	0	3	
	opr	sync	13	11	
	ldl	0	6	0	miles := 0
	stw	0	1	0	
L4:					
	opr	pause	13	9	
	nfix	1	15	1	
	j	L2	8	0	

COMMUNICATION BETWEEN PROCESSES ON DIFFERENT MICROCOMPUTERS

A network of interconnected microcomputers is shown in FIG. 11 in which four microcomputers are illustrated. It will be understood that the network may be extended in two or three dimensions as required. Each of the microcomputers is of similar structure and is interconnected with the serial link of another microcomputer by two unidirectional wires 185 and 186 each of which extends between the output pin 27 on one microcomputer and the input pin 26 of another microcomputer. The wires 185 and 186 are each used solely for these two pin to pin connections and are not shared by other microcomputers or memory connections. Communication between processes in different microcomputers is effected in generally similar manner using an identical sequence of synchronize operations and this will be described with reference to FIGS. 2, 11, 12 and 13. In place of the channel 40 (FIG. 2), a serial link has an input channel 45 and an output channel 46 each consisting of a process register 47 and data register 48 which can be addressed in the same way as the word locations for the memory channels 40 to 43. They are however operated by control logic 50 which will be described further with reference to FIGS. 15 and 16.

When data is transmitted through serial links between two microcomputers, it is in the form of a series of data strings transmitted serially in the form of packets as shown in FIGS. 13*a* and 13*b*. A data packet is transmitted by an output pin 27 to an input pin 26 and has the form shown in FIG. 13*a*. It starts with two successive bits of value 1 followed by 16 data bits and a final stop bit of value 0. An acknowledge packet, as shown in FIG. 13*b* is sent from the output pin 27 of a microcomputer receiving a data packet to the input pin 26 of the microcomputer which sent the data packet. The acknowledge packet consists of a start bit of value zero followed by a stop bit of value 0. The output control logic of each serial link arranges for each output pin 27 to transmit bits of value 0 continuously when it is not sending data or acknowledge packets and consequently the input control logic ignores all signals on the input pins 26 until it receives a 1 start bit of a packet.

When the process register of the input or output channel holds the workspace pointer (WPTR) of a process, the control logic 50 is able to generate requests (called input or output requests) to the CPU (12) for the CPU 12 to schedule the process by adding its workspace pointer to the list awaiting execution. The sync logic 10 provides a selector which is used by the CPU 12 to examine each of the request signals from the serial links in turn. Whenever an active process is descheduled by execution of the "wait" procedure the CPU 12 looks to see if there are any requests from a serial link. If there are several external requests, the CPU 12 services all of them in sequence before executing the next process on the list. The CPU 12 services any requests by scheduling the process held in the process register of the channel which generated the request, and resetting the process register 47 to NIL. The process register 47 of the input or output channels in a link 25 contains the special value READY when that channel is ready to perform communication. The sync operation will cause the procedure "run" which detects the special value READY and instead of scheduling a process, activates the control logic 50 in the link. The control logic in a link may perform a synchronize operation on a channel. The synchronize operation tests the process location of the channel. If the value is NIL, it replaces the value with the special value READY and waits until a sync operation caused by a process instruction on the process register resets the value to NIL. Otherwise, it generates a request to the CPU 12 to schedule the process in the process register as described above, and the CPU then resets the value of the process register to NIL. As a result, a process may use the sync operation to synchronize with the control logic 50 in a link 25 in the same way as it is used to synchronize with another process.

The output control logic 50 in a link 25 first synchronizes with a process using the process register in the output channel, then transmits data in data packets from the data register in the output channel via the output pin 27 (FIGS. 2 and 11), then waits for an acknowledge packet signal on the input pin 26, then synchronizes with the process again using the process register in the output channel. The output control logic 50 performs this operation repeatedly. The input control logic in a link first waits for data from the input pin 26 to arrive in the data register in the input channel, then synchronizes with a process using the process register in the input channel, then synchronizes again with the process using the process register in the input channel, then transmits the acknowledge packet signal to the output pin 27. The input control logic performs this operation repeatedly.

In the following, it is assumed that a process x operated by microcomputer 1 in FIG. 11 wishes to output data through a serial link to a process y operated by

microcomputer 2. To effect this output, the process x stores the data to be output in the data register 187 of the output channel and executes a sync operation on the process register 188 to cause the serial link to start transmission of the data through the pin 27. The process then executes a further sync operation on the same process register 188 to wait until an acknowledge packet is received through the input pin 26 of microcomputer 1. The acknowledge packet signifies that the process y operated by microcomputer 2 has input the data. To input, the process y executes a sync operation on the process register 189 of the input channel of microcomputer 2 to wait for the data packet to arrive from the pin 26 of the microcomputer 2. It then takes the data from the data register 190 and executes a further sync operation to cause the acknowledge signal to be transmitted from the output pin 27 of microcomputer 2.

FIG. 12 shows sequentially the contents of the process registers 188 and 189 during a typical sequence of operations occurring when the processes x and y communicate via the serial link. Reference numerals 188*a–e* represent successive states of the contents of the process register 188 and reference numerals 189*a–e* similarly represent successive states of the contents of the process register 189. First, process x addresses the output channel of microcomputer 1 and loads the data to be output to the data register 187 and performs a sync operation on the output process register 188. Assuming that the process register 188 contains the special value READY 188*a*, indicating that the serial link is ready to output, the sync operation resets the value of the process register 188 to NIL 188*b*. As a result the control logic causes the data from the data register 187 to be transmitted via the single wire connection 185 to the input data register 190 in the microcomputer 2. Provided that process y is not yet waiting for the input, the control logic in microcomputer 2 changes the value of the process register 189 from NIL 189*a* to READY 189*b*, indicating that the data has been received. Process y then executes a sync operation on the process register 189, which has the effect of changing the value of the process register from READY 189*b* to NIL 189*c*. Assuming that microcomputer 2 is ready to transmit an acknowledge signal to microcomputer 1, the control logic changes the value of process register 189 back to READY 189*d*. Process y then takes the data from the data register 190 of the input channel and executes a further sync operation on the process register 189. This resets the process register 189 to NIL 189*e*. As a result the control logic transmits an acknowledge signal through the single wire connection 186. This acknowledge signal is received by the input pin 26 of the microcomputer 1 operating process x. Assuming that process x executes a second sync operation before the acknowledge signal is received, process x is descheduled by the procedure "wait", and its workspace pointer is stored in the process register 188(188*c*). When the acknowledge packet is received the control logic of the serial link generates an request to the CPU of microcomputer 1 to schedule process x. This request is serviced by the CPU of microcomputer 1 as soon as the current process is descheduled and the CPU adds process x to the end of the list and resets the process register to NIL (188*d*). The control logic now resets the process register to READY (188*e*), thereby indicating that the link is ready for a further output. The state of the serial links is now the same as it was before the communication took place, as shown in the sequence of FIG. 12, ready for the next communication.FIG. 14 illustrates the operation on two separate microcomputers of the processes previously described with reference to FIG. 11. In this case however the workspace 171 for counting rotations is on a microcomputer 191 whereas the workspace 172 for counting miles is on a separate microcomputer 192. The two microcomputers 191 and 192 are interconnected through respective serial links 25. Similar reference numerals are used in FIGS. 14 and 10 for similar parts. The only change is that channel "mile" 176 in FIG. 10 is replaced in FIG. 14 by a channel "mile" 176*a* forming an output channel of a serial link in microcomputer 191 and channel "mile" 176*b* forming an input channel of a serial link in microcomputer 192. The sequence of instructions and program used to operate the two processes in FIG. 14 are generally similar to those already described for FIG. 10 except that the address of channel "mile" used by each of the processes will now be the address of a channel of a serial link rather than a channel in memory.

DESCRIPTION OF LINK CONTROL LOGIC

The control logic 50 (FIG. 2) for each of the input and output channels of the serial links will now be described in further detail with reference to FIGS. 15 and 16 in which FIG. 15 shows the control logic for the output channel 46 and FIG. 16 shows the control logic for the input channel 45.

To output, the control logic 50 (FIG. 2) of a link first synchronize with a process using the output process register 47 FIG. 15), then transmits the data from the output data register 48 to the pin 27, then waits for the acknowledge signal from the pin 26, then synchronizes with a process again using the output process register 47. The control logic 50 performs this operation repeatedly.

To input, the control logic 50 (FIG. 2) of a link first waits for data to arrive from the input pin 26 and transfers it to the input data register 48, then synchronizes with a process using the input process register 47 (FIG. 16), then synchronizes again with the process using the input process register, then transmits the acknowledge signal to the pin 27. The control logic 50 performs this operation repeatedly.

The values taken by the output and input process registers 47 may be NIL indicating that neither a process nor the control logic is waiting to synchronize, READY indicating that the control logic is waiting to synchronize, or it may be the workspace pointer of a process waiting to synchronize.

In a link, each process register 47 and each data register 48 is connected to the bus 16 through an address decoder 193. The bus 16 incorporates signal lines for the address, data, and control. Control includes a "write" signal, a "read" signal and a "busy" signal. The "busy" signal is used to ensure that both the CPU and the link control logic do not attempt to change the value of the process register simultaneously.

Each process register 47 in a link incorporates logic 194 to detect if the value in the process register is READY, NIL or a workspace pointer.

The output data register 48 (FIG. 15) is connected to the output pin 27 through an AND gate 195 and an OR gate 196. The input data register 48 (FIG. 16) is connected directly to the input pin 26.

Associated with each process register in a link is a request latch 197 which may be tested by the CPU. Whenever the CPU performs a WAIT procedure, the state of all request latches is tested. If a request latch is

set, the process whose workspace pointer is held in the corresponding process register is scheduled by adding its workspace pointer to the end of the list. The request latch is cleared whenever the CPU writes to the process register.

The input and output of data through the link is controlled by four state machines 282, 283, 284 and 285. Each state machine consists of a state register to hold the current state, and a programmable logic array. The programmable logic array responds to the value of the state register and the input signals to the state machine, and produces a predetermined pattern of output signals and a new value for the state register. A counter 286 is used to count bits as they are transmitted through the link, and a further counter 287 is used to count bits as they are received through the link.

The input and output channel control and data state machines have the following inputs and outputs, wherein the name of the input or output indicates the purpose of the signal.

reference numeral	signal name	purpose
	OUTPUT CONTROL STATE MACHINE 285 (FIG. 15)	
inputs:		
200	Mbusy	Memory bus busy
201	Reset	Transputer reset
202	Pregready	Process Register = READY
203	Pregnil	Process Register = NIL
204	Pregwptr	Process Register holds a workspace pointer
205	Datagone	Data transmitted from output data register
264	Ackready	Acknowledge received by input state machine
outputs:		
210	Setrequest	Set cpu request
211	Datago	Initiate data transmission
212	SetPregready	Set Process Register to READY
213	SetPregnil	Set Process Register to NIL
265	Acktaken	Confirm receipt of acknowledge
	OUTPUT DATA STATE MACHINE 284 (FIG. 15)	
inputs:		
201	Reset	Transputer reset
211	Datago	Initiate data transmission
220	Countzero	Test if bit count zero
261	Ackgo	Initial acknowledge transmission
outputs:		
221	Loadcount	Set Bit Counter to number of bits to be transmitted
222	Deccount	Decrease bet counter by one
223	Oneout	Set output pin to one
224	Dataout	Set output pin to lease significant bit of shift register
225	Shiftout	Shift data register one place
205	Datagone	Transmission of data complete
260	Ackgone	Transmission of acknowledge complete
	INPUT CONTROL STATE MACHINE 283 (FIG. 16)	
inputs:		
200	Mbusy	Memory bus busy
201	Reset	Transputer reset
262	Dataready	Data received from pin
242	Pregready	Process Register = READY
243	Pregnil	Process Register = NIL
244	Pregwptr	Process Register holds a workspace pointer
260	Ackgone	Transmission of acknowledge complete
outputs:		
220	Setrequest	Set cpu request
222	SetPregready	Set Process Register to READY
222	SetPregnil	Set Process Register to NIL
261	Ackgo	Initiate acknowledge transmission

-continued

reference numeral	signal name	purpose
263	Datataken	Confirm receipt of data
	INPUT DATA STATE MACHINE 282 (FIG. 16)	
inputs:		
201	Reset	Transputer reset
230	Datain	Data from pin
231	Countzero	Test if bit count zero
outputs:		
240	Loadcount	Set Bit Counter to number of bits to be received
241	Deccount	Decrease bit counter by one
244	Shiftin	Shift data register one place taking least significant bit from pin
245	Setdataready	Reception of data complete
246	Setackready	Reception of acknowledge complete

The sequences of each state machine are set out below with reference to present state, next state, input and output of each machine.

In any state, the outputs listed under the "outputs" column are one, and all other outputs are zero. All inputs are ignored except those mentioned in the "inputs" column. The symbols /, / and Δ are used to denote the boolean operations and, or and not respectively.

State	Inputs		Outputs	Next state
	OUTPUT CONTROL STATE MACHINE 285			
any	Reset		SetPregnil	sync1
sync1	Mbusy			syc1
sync1	(ΔMbusy) /	Pregnil	SetPregready	syncreq1
sync1	(ΔMbusy) /	Pregwptr	Setrequest	syncreq1
syncreq1	ΔPregnil			syncreq1
syncreq1	Pregnil			send1
send1	ΔDatagone		Datago	send1
send1	Datagone			send2
send2	Datagone			send2
send2	ΔDatagone			waitack1
waitack1	ΔAckready			waitack1
waitack1	Ackready			waitack2
waitack2	Ackready		Acktaken	waitack2
waitack2	ΔAckready			sync2
sync2	Mbusy			sync2
sync2	(ΔMbusy)/	Pregnil	SetPregready	syncreq2
sync2	(ΔMbusy)/	Pregwptr	Setrequest	syncreq2
syncreq2	ΔPregnil			syncreq2
syncreq2	Pregnil			sync1
	OUTPUT DATA STATE MACHINE 284			
any	Reset			idle
idle	(ΔDatago)/	(ΔAckgo)		idle
idle	Ackgo		Oneout	ackflag
idle	(ΔAckgo)/	Datago	Oneout	dataflag
ackflag				ackend
dataflag			Oneout Loadcount	databits
databits	ΔCountzero		DecCount Shiftout Dataout	databits
databits	Countzero			dataend
dataend	Datago		Datagone	dataend
dataend	ΔDatago			idle
ackend	Ackgo		Ackgone	ackend
ackend	ΔAckgo			idle
	INPUT CONTROL STATE MACHINE 283			
any	Reset		SetPregnil	receivel
receivel	ΔDataready			receivel
receivel	Dataready			sync1
sync1	Mbusy			sync1
sync1	(ΔMbusy)/	Pregnil	SetPregready	syncreq1
sync1	(ΔMbusy)/	Pregwptr	Setrequest	syncreq1
syncreq1	ΔPregnil			syncreq1
syncreq1	Pregnil			sync2

-continued

State	Inputs		Outputs	Next state
sync2	Mbusy			sync2
sync2	(ΔMbusy)/	Pregnil	SetPregready	syncreq2
sync2	(ΔMbusy)/	Pregwptr	Setrequest	syncreq2
syncreq2	ΔPregnil			syncreq2
syncreq2	Pregnil			receive2
receive2	Dataready		Datataken	receive2
receive2	ΔDataready			acksend1
acksend1	ΔAckgone		Ackgo	acksend1
acksend1	Ackgone			acksend2
acksend2	Ackgone			acksend2
acksend2	ΔAckgone			receive1
	INPUT DATA STATE MACHINE 282			
any	Reset			idle
idle	ΔDatain			idle
idle	Datain			start
start	ΔDatain		SetAckready	idle
start	Datain		LoadCount	databits
databits	ΔCountzero		Shiftin DecCount	databits
databits	Countzero		Shiftin	dataend
dataend	—		SetDataready	idle

As shown in FIG. 16, the input control logic includes a flip-flop 280 connected to the output 246 of the input data state machine 282. A further flip-flop 281 is connected to the output 245 of the input data state machine 282. Both control state machines are controlled by clock pulses derived from the clock 28. For some of the links, both data state machines are also controlled by clock pulses derived from the clock 28. For the link shown in FIGS. 15 and 16, the data state machines are controlled by clock pulses derived from a different clock 22 related in phase to clock 28, which allows this link to operate at a lower speed. Two different clock frequencies can be obtained in order to achieve maximum efficiency depending on the type of microcomputer network which is operated. When microcomputers are grouped closely together communications between them can be carried out more quickly in which case a higher clock frequency can be used. A lower clock frequency can be used to enable satisfactory communication where the microcomputers are more remote and require a lower operating speed.

In both the input and output channels the control state machine monitors the content of the process register 47 and when appropriate generates a CPU request on line 199 by setting the latch 197.

The output control state machine 285 first synchronizes with a process using the output process register 47. It then uses the "datago" signal 211 to cause the output data state machine 282 to output the data in the output data register 48 through the pin 27. The output data state machine 284 sends the data in the manner described with reference to FIG. 13*a* and shifts the data in the register 48 until a count in the counter 286 expires. When it has done this it returns the "datagone" signal 205 to the output control state machine to indicate that the transfer of data is complete and that the "datago" signal should be removed. The output control state machine then waits for the "ackready" signal 264 from the latch 280, signifying that the input data state machine 282 has received an acknowledge packet as described in FIG. 13*b* from the pin 26. In response to the "ackready" signal 264, the output control state machine outputs an "acktaken" signal 265, which resets the latch 280. The output control logic then uses the output process register 47 to synchronize again with the outputting process.

The input data state machine 282 and the microcomputer at the other end of the link is waiting for "start bit" to appear on the input pin 26. When a data packet is detected, the input data state machine 282 of that microcomputer shifts data into the data shift register 48 until the counter 287 indicates that the appropriate number of bits have been received, and then sets the "data received" latch 281. The input control state machine 283 detects the "dataready" signal 262 and responds by resetting the "data received" latch 281. It then synchronizes with an inputting process using the input process register 47. It then synchronizes again with the inputting process using the process register 47 to confirm that the process has taken the data from the data register 48, and then uses the "ackgo" signal 261 to cause the output data state machine to transmit an acknowledge packet via the pin 27. When the output data state machine 284 is not transmitting data it generates the start and stop bits which constitute the acknowledge packet described in FIG. 13*b*. The input data state machine 282 of the microcomputer which transmitted the data packet detects the acknowledge packet and sets the "acknowledge received" latch 280. As described above, the output control state machine 285 of the transmitting microcomputer has been waiting for this and on detecting the signal resets the latch 280 and performs a second synchronize operation. The state of the link logic in both the output and input links is now the same as it was before the communication took place so that it is ready for the next transmission.

CHIP AND MEMORY FORMATION

As mentioned above, the microcomputer of this example is particularly advantageous in having sufficient memory in the form of RAM on the chip (integrated circuit device) to enable the microcomputer to operate without the necessity for external memory, although external memory can be used when required. There are a number of problems in providing sufficient space for adequate memory on the same chip as the processor. It is necessary to minimise the area required for each memory cell as well as reducing noise interference in the RAM from asynchronously operating circuitry such as a processor on the same chip, while at the same time providing a satisfactory manufacturing yield of acceptable microcomputers from a number of silicon chips, particularly as the memory may be the largest and densest component created on the chip.

In order to minimize the chip area required for each memory cell, this example uses static RAM cells (SRAM) using high impedance resistive loads rather than the more conventional depletion transistor loads or complementary pull-up transistors. The manufacturing technology used in this example employs a film of high resistivity polycrystalline silicon in which the resistive loads are formed. The memory may have 32 K bits of SRAM where each cell consists of transistors having gates formed in a film of polycrystalline silicon. The transistor gates and resistive loads may be formed in the same, or different films of polycrystalline silicon.

Resistor load SRAMs are susceptible to interference from electrical noise injected into the silicon material in which they are formed and stored data can be corrupted by any minority carriers which may be present. In order to shield the SRAM from noise generated by other on chip circuitry and from minority carriers injected by other on chip circuitry the SRAM is formed in an electrically isolated area of silicon as shown in FIG. 17. An

n-channel substrate 300 is formed with separate p-wells 301, and 302. The array of RAM cells are isolated from other circuitry and associated substrate noise by locating the RAM array in the p-well marked 301. This isolates the RAM cells from minority carriers generated in the substrate by the well-to-substrate potential barrier and any minority carriers generated within the well have a high probability of being collected in the substrate. In FIG. 17, the RAM array will be an n-channel array located in the p-well 301. Any n-channel transistors of peripheral circuitry are isolated from the RAM array by placing them in a further p-well 302.

This technique is fully compatible with either NMOS or P-well CMOS manufacturing technology. In the current example P-well CMOS is used any any p-channel transistors of peripheral circuitry are placed on the n-substrate and isolated from the RAM array by the well-to-substrate potential barrier. Each well containing a memory array is surrounded by a metal ground which contacts the memory array well around its periphery via a heavily doped p diffusion. Within the memory array there is a p diffusion contacting the well to ground for each pair of cells. Substrate bias is unnecessary.

In order to provide acceptable manufacturing yield of products from silicon chips, memory redundancy is incorporated. The memory is divided into rows and columns accessible respectively by row and column decoders. In addition to the normal rows and normal columns the redundancy provides some additional rows and columns together with spare row and column decoders in order to obtain access to the spare rows and columns. The spare column decoders and spare row decoders each incorporate fuses which for example can be open circuited by use of a laser so that when any defective rows or columns are determined during test, fuses can be open circuited by laser techniques to disable the row or column detector of any normal rows or columns which have been found to be defective and the replacement row or column from the redundant rows and columns can be brought into an enabled position by programming the appropriate spare row decoder or spare column decoder with the address of the defective row or column.

In order to allow N-well CMOS manufacturing technology to be used the following alternative isolation technique may be employed. Referring to FIG. 18 a low resistivity P type substrate (405) is used on which a high resistivity P type epitaxial layer is formed.

The cell array is formed in this epitaxial layer in region (401) and is entirely surrounded by a deep N-well diffusion (402). Minority carriers generated by other circuitry in region (403) will be attracted to the N-wells (402) where they become harmless majority carriers, or will recombine in the heavily doped P-type substrate (405). P-channel transistors are placed in N-wells (404) where they are isolated by the well to substrate potential barrier.

ADDITIONAL MATERIAL

The invention is not limited to the details of the foregoing example. For instance, although the serial links shown in FIG. 2 have separate process registers 47, the function provided by the registers 47 may be effected by memory locations in the RAM 19. In this case the CPU must be able to identify the serial link which it is serving and this may be achieved by connecting each channel of each serial link separately to the sync logic 10 in FIG. 2.

One set of data registers and buses is shown in FIG. 3 and in some cases it may be desirable to include two such sets in one microcomputer, or even to have two CPUs in one microcomputer.

The principle described above of using pfix and nfix functions to vary the length of operand is applicable to a microcomputer of any word length.

The invention is not limited to a machine operating with 16 bit words nor to 16 bit operand registers, e.g. processors having a word length of 8 bits or multiples of 8 bits may use these instructions. The invention is particularly applicable to 32 bit word microcomputers.

The CPU may include further registers, in an evaluation stack, such as a CREG or even DREG in addition to the A and B registers. Some functions and operations may then be modified to allow the additional registers. For example:

Areg:=Breg may be replaced by SEQ
Areg:=Breg
Breg:=Creg
Creg:=Dreg
Breg:=Areg may be replaced by SEQ
Dreg:=Creg
Creg:=Breg
Breg:=Areg

Other functions or operations may of course be added to exploit the extra registers. Although the illustrated embodiment described herein and shown in FIG. 3 includes only an A register and a B register, in a preferred embodiment of the present invention, three registers are used in a stack.

It will be appreciated that in the above description, the function set lists a plurality of functions followed by an extendable list of operations which may be selected by use of the indirect function "operate". In all cases these functions and operations can be considered as forms of instruction usable in the program to operate the microcomputer. However in order to obtain the advantages discussed above for a fixed format of "instruction" as shown in FIG. 5, the list of functions and operations can be considered as a set of primary instructions (consisting of the direct functions, prefixing functions and indirect functions) and a set of secondary instructions (consisting of the operations which may be selected by use of the indirect function). To maximize efficiency, the primary instructions which are most commonly used require only 4 bits of the instruction format shown in FIG. 5 and so the other 4 bits can be used for data to be loaded into the operand register 65 and used as an operand for the instructions. For the secondary instructions which are less commonly used, all 8 bits of the instruction format shown in FIG. 5 are needed to identify the instruction required.

Consequently the fixed format of the instruction shown in FIG. 5 allows no data to accompany a secondary instruction and secondary instructions therefore operate on data held in registers other than the operand register 65.

Although the instruction format shown in FIG. 4 comprises 8 bits divided into two halves, it will be understood that other bit lengths may be used and the division into function and data need not necessarily provide equal bit lengths for the two parts.

It is to be appreciated that the present arrangement described herein provides a combination which dramatically improves the efficiency and throughput of the microcomputer. By using instructions having a constant format, by having a function set where the most often

used functions are directly available whereas other functions are indirectly available, by arranging for communication between processes and synchronization among them, by permitting point-to-point communication between microcomputers, and by providing memory on the same chip as each microprocessor, a microcomputer according to various aspects of the invention can achieve a speed of 10 million instructions per second. An array housed on a board of only 10 inches by 20 inches should be able to achieve a speed of 1000 million instructions per second. A Transputer (trade mark) microcomputer array using OCCAM (trade mark) should be able to achieve speeds approximately two orders of magnitude faster, than, for example, a Motorola 68000 using PASCAL. A single Transputer programmed in OCCAM should be about two or three times faster than a single 68000 microprocessor using PASCAL. In the prior art, when microcomputers are added in an array, the incremental gain in performance is progressively less and less with an increase in processors. However, by using the microcomputer of this example, the increase in performance is a linear function of the number of processors. Thus it will be appreciated that the present combination achieves dramatically increased performance over the state of the art.

We claim:

1. A microcomputer comprising an on-chip processor and an on-chip memory on a single integrated circuit chip having a substrate of semiconductor material of a first type, wherein said on-chip memory comprises a high density RAM array having at least 1K bytes for holding a sequence of instructions for execution by said on-chip processor, said microcomputer including:

(a) an instruction pointer circuit for addressing said RAM to obtain program instructions therefrom,

(b) an instruction receiving circuit coupled to said RAM for receiving said instructions from said program stored in said RAM,

(c) an instruction decoder circuit coupled to said instruction receiving circuit for decoding instructions received by said instruction receiving circuit,

(d) a plurality of on-chip transistors comprising circuitry operable independently of the operation of said RAM,

(e) a first isolation region in said substrate, said first isolation region being of the same type of material as that of said substrate and containing all of said memory cells of said high density RAM array, and

(f) a second isolation region separate from said first isolation region and being of the same type of material as that of said substrate, said second isolation region containing some of said transistors which are operable independently of said operation of said RAM,

(g) isolation means formed in said substrate for isolating said first and second regions,

whereby said high density RAM is located on the same chip as said independently operation transistors and is protected from noise due to independent operation of said transistors.

2. A microcomputer according to claim 1 wherein said isolation means is a well.

3. A microcomputer according to claim 1 wherein said isolation means is a well having a conductivity opposite to that of said substrate.

4. A microcomputer according to claim 1 wherein said first isolation region comprises a plurality of isolation regions each containing a portion of said memory cells of said RAM array.

5. A microcomputer according to claim 1 wherein said substrate includes an epitaxial layer and said memory cells are located in said epitaxial layer.

6. A microcomputer comprising an on-chip processor and an on-chip memory on a single integrated circuit chip having a substrate of semiconductor material of a first type, wherein said on-chip memory comprises a high density RAM array having at least 1K bytes for holding a sequence of instructions for execution by said on-chip processor, said microcomputer including:

(a) an instruction pointer circuit for addressing said RAM to obtain program instructions therefrom,

(b) an instruction receiving circuit coupled to said RAM for receiving said instructions from said program stored in said RAM,

(c) an instruction decoder circuit coupled to said instruction receiving circuit for decoding instructions received by said instruction receiving circuit,

(d) a plurality of on-chip transistors comprising circuitry operably independently of the operation of said RAM,

(e) a first isolation well formed in said substrate of a semiconductor material of different type of material than said substrate and defining a first isolation region in said substrate, said first isolation region being of the same type of material as said substrate,

(f) a second isolation well formed in said substrate of a semiconductor material of different type than said substrate, said first isolation region or said second well containing all of said memory cells of said high density RAM array, the other containing some of said transistors which are operable independently of said operation of said RAM,

whereby said high density RAM is located on the same chip as said independently operating transistors and is protected from noise due to independent operation of said transistors.

7. A microcomputer according to claim 6 wherein said isolation well surrounds said first isolation region.

8. A microcomputer according to claim 6 wherein the first of said isolation regions comprises a plurality of isolation regions each containing a portion of said memory cells of said high density RAM array.

9. A microcomputer according to claim 6 wherein said substrate includes an epitaxial layer and said memory cells are located in said epitaxial layer.

10. A microcomputer comprising an on-chip processor and an on-chip memory on a single integrated circuit chip having a substrate of semiconductor material of a first type, wherein said on-chip memory comprises a high density RAM array having at least 1K bytes for holding a sequence of instructions for execution by said on-chip processor, said microcomputer including:

(a) an instruction pointer circuit for addressing said RAM to obtain program instructions therefrom,

(b) an instruction receiving circuit coupled to said RAM for reeving said instructions from said program stored in said RAM,

(c) an instruction decoder circuit coupled to said instruction receiving circuit for decoding instructions received by said instruction receiving circuit,

(d) a plurality of on-chip transistors comprising circuitry operable independently of the operation of said RAM,

(e) a first isolation region in said substrate, said first isolation region containing all of said memory cells of said high density RAM array, and

(f) a second isolation region in said substrate separate from said firs isolation region, said second isolation region containing some of said transistors which are operable independently of said operation of said RAM, said firs and second regions noise isolated from each other,

whereby said high density RAM is located on the same chip as said independently operating transistors and is protected from noise due to independent operation of said transistors.

11. A microcomputer according to claim 10 wherein said first isolation region comprises a plurality of isolation regions each containing a portion of said memory cells of said RAM array.

12. A microcomputer according to claim 10 wherein said substrate includes an epitaxial layer and said memory cells are located in said epitaxial layer.

13. A microcomputer comprising an on-chip processor and an on-chip memory on a single integrated circuit chip having a substrate of semiconductor material of a first type, wherein said on chip memory comprises a high density RAM array having at lest 1K bytes for holding a sequence of instructions for execution by said on-chip processor, said microcomputer including:

(a) an instruction pointer circuit for addressing said RAM to obtain program instructions therefrom,

(b) an instruction receiving circuit coupled to said RAM for receiving said instructions from said program stored in said RAM,

(c) an instruction decoder circuit coupled to said instruction receiving circuit for decoding instructions received by said instruction receiving circuit,

(d) a plurality of on-chip transistors comprising circuitry operable independently of the operation of said RAM,

(e) a first isolation well or wells formed in said substrate of the same type of semiconductor material as said substrate,

(f) a second isolation well or wells formed in said substrate separate from said first isolation well or wells and formed of a semiconductor material of different type than said substrate, all of said memory cells of said high density RAM array being contained in either said first or said second isolation well or wells, some of said transistors which are operable independently of the operation of said RAM being contained in the other of said first or second isolation well or wells,

whereby said high density RAM is located on the same chip as said independently operating transistors and is protected from noise due to independent operation of said transistors.

14. A microcomputer according to claim 13 including a third isolation well formed around said first isolation well or wells, said third isolation well being of a different type of material than that of said substrate.

15. A microcomputer according to claim 13 wherein said first isolation well comprises a plurality of isolation wells each containing a portion of said memory cells of said RAM array.

16. A microcomputer according to claim 13 wherein said substrate includes an epitaxial layer and said memory cells are located in said epitaxial layer.

17. A microcomputer comprising an on-chip processor and an on-chip memory on a single integrated circuit chip having a substrate of semiconductor material of a first type, wherein said on-chip memory comprises a high density RAM array having at least 1K bytes for holding a sequence of instructions for execution by said on-chip processor, said microcomputer including:

(a) an instruction pointer circuit for addressing said RAM to obtain program instructions therefrom,

(b) an instruction receiving circuit coupled to said RAM for receiving said instructions from said program stored in said RAM,

(c) an instruction decoder circuit coupled to said instruction receiving circuit for decoding instructions received by said instruction receiving circuit,

(d) a plurality of on-chip transistors comprising circuitry operable independently of the operation of said RAM,

(e) a first isolation well or wells formed in said substrate of a semiconductor material having the same conductivity type but different resistivity than said substrate, (f) a second isolation well or wells formed in said substrate separate from said first isolation well or wells and formed of a semiconductor material having the same conductivity type but different resistivity than said substrate, either of said first or said second isolation well or wells containing all of said memory cells of said high density RAM array, the other of said first or said second isolation well or wells containing some of said transistors which are operable independently of said operation of said RAM,

whereby said high density RAM is located on the same chip as said independently operating transistors and is protected from noise due to independent operation of said transistors.

18. A microcomputer according to claim 17 wherein the one of said first or said second isolation wells comprises a plurality of wells each containing a portion of said memory cells of said RAM array.

19. A microcomputer according to claim 17 wherein said substrate includes an epitaxial layer and said memory cells are located in said epitaxial layer.

20. A microcomputer comprising an on-chip processor and on-chip memory on a single integrated circuit chip having a substrate of semiconductor material of a first type, wherein said on-chip memory comprises a high density RAM array having at least 1K bytes for holding a sequence of instructions for execution by said on-chip processor, said microcomputer including:

(a) an instruction pointer circuit for addressing said RAM to obtain program instructions therefrom,

(b) an instruction receiving circuit coupled to said RAM for receiving said instructions from said program stored in said RAM,

(c) an instruction decoder circuit coupled to said instruction receiving circuit for decoding instructions received by said instruction receiving circuit,

(d) a plurality of on-chip transistors comprising circuitry operable independently of the operation of said RAM,

(e) a first isolation well or wells formed in said substrate of a semiconductor material of a different type than said substrate,

(f) a first region formed in said substrate, said first region defined by said first isolation well or wells, said first region being of the same type of semiconductor material as said substrate of said memory cells, all of said high density RAM array being contained in either said first isolation well or wells

or in said first region, some of said transistors which are operable independently of said operation of said RAM being contained in the other of said first isolation well or wells or in said first region, whereby said high density RAM is located on the same chip as said independently operating transistors and is protected from noise due to independent operation of said transistors.

21. A microcomputer according to claim 20 wherein the one of said first well or wells and said first region comprises a plurality of wells or regions each containing a portion of said memory cells of said RAM array.

22. A microcomputer according to claim 20 wherein said substrate includes an epitaxial layer and said memory cells are located in said epitaxial layer.

23. A microcomputer comprising an on-chip processor and an on-chip writable memory on a single integrated circuit chip having a substrate of semiconductor material of a first type, wherein said on-chip writable memory comprises a high density memory array having at least 1K bytes for holding a sequence of instructions for execution by said on-chip processor, said microcomputer including:

(a) an instruction pointer circuit for addressing said memory array to obtain program instructions therefrom,

(b) an instruction receiving circuit coupled to said memory array for receiving said instructions from said program stored in said memory array,

(c) an instruction decoder circuit coupled to said instruction receiving circuit for decoding instructions received by said instruction receiving circuit,

(d) a plurality of on-chip transistors comprising circuitry operable independently of the operation of said memory array,

(e) a first isolation region in said substrate, said first isolation region containing all of said memory cells of said high density memory array, and

(f) a second isolation region in said substrate separate from said first isolation region, said second isolation region containing some of said transistors which are operable independently of said operation of said memory array, said first and second regions noise isolated from each other,

whereby said high density memory array is located on the same chip as said independently operating transistors and is protected from noise due to independent operation of said transistors.

24. A microcomputer according to claim 23 wherein said first isolation region comprises a plurality of isolation regions each containing a portion of said memory cells of said memory array.

25. A microcomputer according to claim 23 wherein said substrate includes an epitaxial layer and said memory cells are located in said epitaxial layer.

* * * * *

US005347185A

United States Patent [19]

Tailliet

[11] Patent Number: 5,347,185

[45] Date of Patent: Sep. 13, 1994

[54] **PROTECTION STRUCTURE AGAINST LATCH-UP IN A CMOS CIRCUIT**

[75] Inventor: Francois Tailliet, Epinay sur Seine, France

[73] Assignee: SGS-Thomson Microelectronics, S.A., Gentilly Cedex, France

[21] Appl. No.: 892,941

[22] Filed: Jun. 3, 1992

[51] Int. Cl.[5] H01P 1/22; H03K 3/26

[52] U.S. Cl. 307/540; 307/566; 307/567; 307/318; 361/91; 361/111

[58] Field of Search 307/566.7, 540, 318; 361/91, 111

[56] References Cited

U.S. PATENT DOCUMENTS

4,032,800	6/1977	Dröschen et al.	307/318
4,922,367	5/1990	Hidaka	361/111
4,939,616	7/1990	Roundtree	361/56
4,948,989	8/1990	Spratt	307/318
5,006,736	4/1991	Davies	307/318
5,196,981	3/1993	Kuo	361/111
5,198,957	3/1993	Welty et al.	361/111

FOREIGN PATENT DOCUMENTS

0324185	7/1989	European Pat. Off. .
9106285	5/1991	France .
2227898	8/1990	United Kingdom .

OTHER PUBLICATIONS

Patent Abtracts of Japan, vol. 11, No. 165 (E-510)(2612) May 27, 1987 (summarizing Japanese app'n JP-A-61,296,770).

Patent Abstracts of Japan, vol. 11, No. 265 (E-535)(2712) Aug. 27, 1987 (summarizing Japanese app'n JP-A-62,069,661).

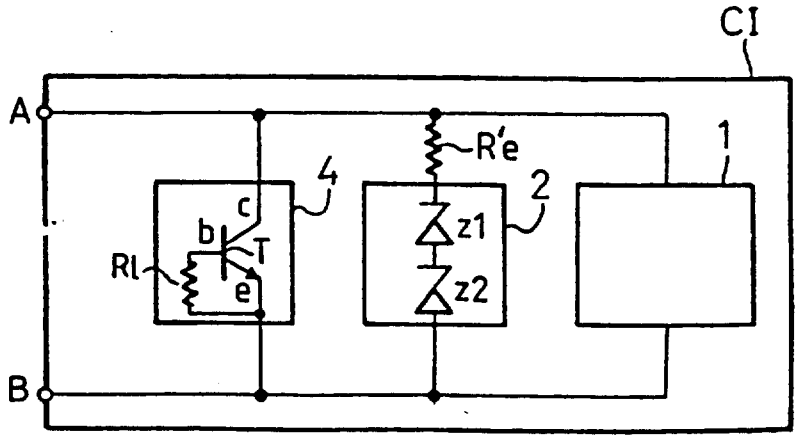
Primary Examiner—Margaret Rose Wambach

Attorney, Agent, or Firm—Robert Groover

[57] **ABSTRACT**

A CMOS circuit protected against latch-up. A limiter parallel to the internal circuitry of the CMOS circuit increases the external current for the triggering of the latch-up in the event of overvoltage on the supply. In one embodiment, the parallel limiter is intrinsically protected against electrostatic discharges. In another embodiment, the limiter is protected by a series connected resistor and a separate shunt-connected ESD protection structure.

24 Claims, 2 Drawing Sheets



Ex B

FIG_1

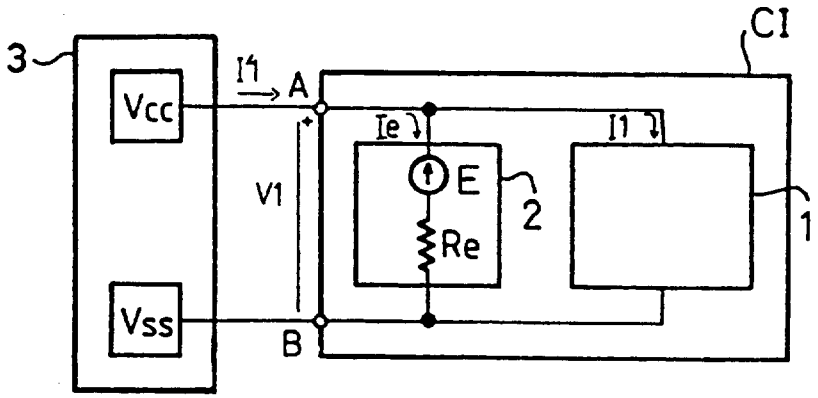


FIG_2

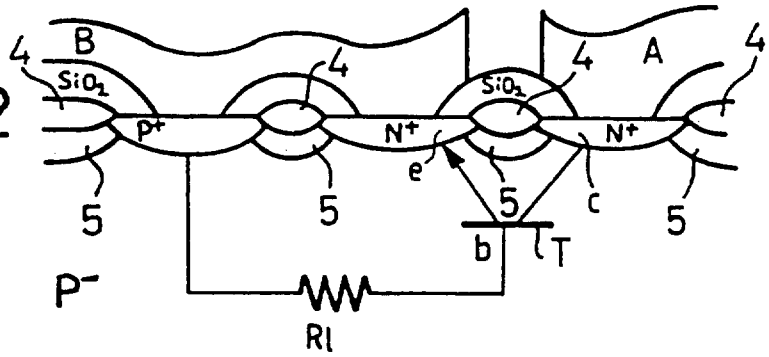


FIG_3

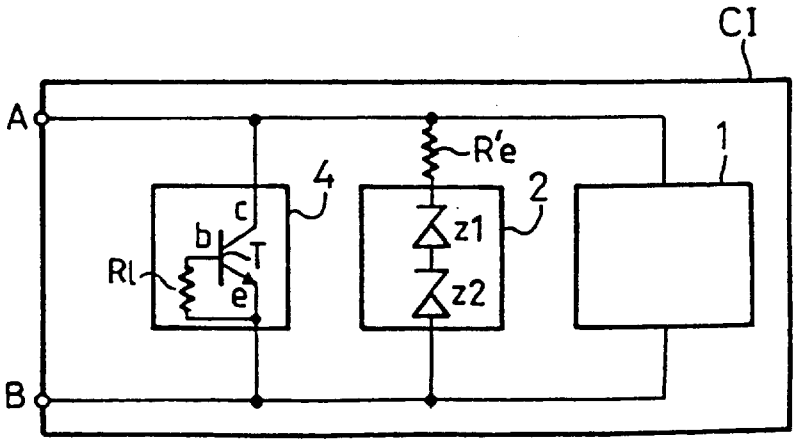


FIG_4

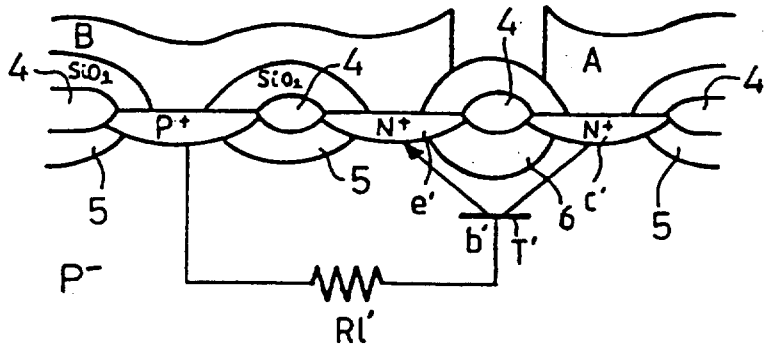
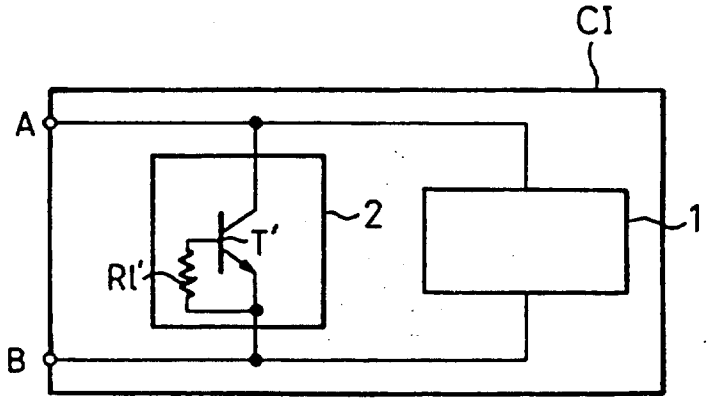


FIG_5



PROTECTION STRUCTURE AGAINST LATCH-UP IN A CMOS CIRCUIT

BACKGROUND OF THE INVENTION

1. Field of the Invention

The invention relates to CMOS integrated circuits and, more specifically, to the protection of these circuits against the phenomenon of latch-up which may destroy certain parts of the CMOS integrated circuit.

CMOS technology gives rise to numerous parasitic structures with four layers PNPN. Although these P nor N regions are needed for the desired transistors, the combination of four successive P,N,P, and N layers produces a (undesired but operable) thyristor. As extensively discussed in the device literature, such a thyristor can be analyzed in terms of two parasitic bipolar transistors and associated series resistors.

A single unwanted pulse or an overvoltage on the supply may prompt the triggering of some of these thyristors: this is the phenomonen of latch-up of CMOS circuits.

2. Description of the Prior Art

The triggering of these parasitic thyristor structures sets up a variably sharp short-circuit which is generally destructive through the over-heating of the circuit. The standard way of preventing latch-up is to use very strict rules for the designing of the CMOS circuits. In particular:

to reduce the gain of the bipolar transistors, the width of their base is increased by giving the wells a substantial depth and by making the diffusions in the substrate at a distance from the wells;

to reduce the lateral resistance values, the substrate and well connectors are greatly increased in number.

The reduction of the lateral resistance values also serves to reduce the gain of the bipolar transistors (by the short-circuit of the emitter-base junction).

These rules entail very severe constraints and are very costly in terms of space with uncertain results in practice: for one circuit, they will give very good results and for another circuit they will give poor results.

In certain cases, it is preferred to use an epitaxial substrate instead of one that is uniformly doped in the bulk (a bulk substrate). The epitaxial substrate is highly enriched, for example with N^+ doping, surmounted by a very fine epitaxial layer with low level N^- doping. It can be used chiefly to reduce the lateral resistance values but this is a very costly technology for a result that is still not sure (because of the problem of the reproducibility of the result from one circuit to another).

Take a CMOS circuit. At two supply terminals, it receives a positive voltage Vcc, generally five volts in terms of nominal value, and a more negative (ground) voltage Vss, generally equal to zero volts. The CMOS circuit accepts a maximum voltage $Vmax = Vcc - Vss$, at its terminals, of the order of 7 volts. The immunity of the CMOS circuit to the latch-up phenomenon is defined by the over-voltage V1 and the external current I1 for which the latch-up is triggered. Typically, it is sought to have V1 greater than 2Vmax and the current I1 as high as possible. The above-described approaches can be used to have I1 of the order of 20 milliamperes. By the present invention, it is sought to take I1 to one hundred milliamperes. Thus, CMOS products will be obtained that are truly well protected against the latch-up phenomenon. Furthermore, it is sought to reduce both the cost of the protection in terms of the place occupied by the circuit and its cost price.

The invention starts with a standard technology CMOS circuit with limit parameters of immunization to latch-up (V1, I1) proper to the internal circuitry.

In the invention, a limiter component is interposed between the supply terminals, in parallel with the internal CMOS circuitry. This limiter component gets triggered for a voltage E at its terminals of less than V1 and consumes high current I_e far higher than the current I1 for a voltage V1 at its terminals, i.e. this is a component with low equivalent resistance when it is triggered.

In the invention, the limiter component is integrated into the CMOS circuit. Thus, while the intensity of the external current needed to prompt the latch-up phenomenon in the internal circuitry remains unchanged (I1), by contrast the total external current I'1 to be given to the CMOS circuit, equal to I1 without the limiter, is increased in the invention by the current (Ie) flowing through the limiter. Thus, the apparent value I'1 of the external current for triggering the latch-up has been increased. This apparent value then depends on the sizing of the limiter.

In one improvement, the limiter is protected against electrostatic discharges.

SUMMARY OF THE INVENTION

The invention therefore relates to a CMOS circuit protected against a latch-up phenomenon, comprising an internal circuitry powered between a first positive supply terminal and a second negative or zero supply terminal, the internal circuitry having, as known parameters, a current and a voltage for the triggering of the latch-up phenomenon, wherein a limiter is placed in parallel on the internal circuitry, between the first supply terminal and the second supply terminal, this limiter having a triggering threshold below the triggering voltage for the latch-up of the internal circuitry and consuming current of a value far higher than the current for triggering the latch-up of the internal circuitry when a voltage at least equal to the latch-up triggering voltage is applied between the first and second supply terminals.

BRIEF DESCRIPTION OF THE DRAWINGS

Other features and advantages of the invention are given in the following description, which is made by way of a non-restrictive indication of the invention and with reference to the appended drawings, of which:

FIG. 1 shows a block diagram of a CMOS circuit comprising a protection device against latch-up according to the invention;

FIG. 2 shows a structure of a known device for protection against electrostatic discharges;

FIG. 3 shows an electrical diagram of a CMOS circuit in a first exemplary embodiment of a limiter according to the invention, using the device shown in FIG. 2;

FIG. 4 shows a structure of a limiter in a second exemplary embodiment according to the invention;

FIG. 5 shows an electrical diagram of a CMOS circuit according to FIG. 4.

MORE DETAILED DESCRIPTION

FIG. 1 is a block diagram of a protection device according to the invention. A CMOS integrated circuit CI has an internal circuitry 1 to be shielded against latch-up and a limiter 2. Each element 1, 2 is parallel-connected to two supply terminals A and B. A supply 3

gives a positive voltage Vcc at the terminal A and a negative or zero voltage Vss at the terminal B.

The potential difference V between the two supply terminals of the CMOS circuit CI has, in a standard way, a typical value of 5 volts referenced Vdd and a maximum value of the order of 7 volts referenced Vmax. For a voltage threshold E at the terminals of the CMOS circuit, hence at the terminals of the limiter, the latter gets triggered. It then behaves like a Thévenin generation of voltage E in series with a low value resistor Re (FIG. 1).

Thus, if the internal circuitry 1 of the CMOS circuit is characterized by a latch-up triggering current/voltage pair referenced (I1, V1), if the voltage V at the terminals of the CMOS circuit becomes equal to V1 (overvoltage on the supply) and if the triggering threshold E of the limiter is lower than the voltage V1, then the intensity of the current I_e that flows through the limiter is given by the following expression:

$$Ie = \frac{V1 - E}{Re}$$

Since the limiter 2 and the internal circuitry 1 are each parallel-connected to the supply terminals A and B of the CMOS circuit, in the event of supply overvoltage V1, the external supply should give a current I'1 equal to the sum of the currents of the two branches, i.e. I'1=Ie+I1 is now the amount required to trigger the latch-up.

The current for triggering the latch-up of the internal circuitry 1 is always I1, but the apparent current I'1 for triggering the latch-up of the CMOS circuit is itself increased by the current flowing through the limiter which has the voltage V1 at its terminals.

It is easy to obtain fairly substantial currents of the order of a hundred milliamperes with well known limiter structures, two examples of which are described below.

To prevent excessive power consumption by the CMOS circuit, preferably a limiter will be chosen with a threshold E greater than the maximum voltage Vmax allowable by the CMOS circuit. In one example Vmax is equal to 7 volts and E is equal to 12 volts.

Generally, the limiter according to the invention has a threshold E ranging from 10 to 15 volts (E>Vmax, $E<V_1$) and a low dynamic resistance Re (up to several tens of ohms) so as to let through a great deal of current: a high gain in latch-up current is sought.

Thus, in a standard example of a CMOS circuit with a latch-up triggering voltage V1 of the order of 18 volts and a latch-up triggering voltage I1 of the order of 20 milliamperes, it is easy to determine the necessary characteristics (E, Re) of the limiter to have an apparent triggering current I'1 of (for example) 320 milliamperes for an overvoltage V1. Indeed, we have the following relationships:

$$Vmax < E < V1$$

$$I'1 = I1 + Ie \text{ and } Ie = \frac{(V1 - E)}{Re}$$

E is chosen for example as being equal to 12 volts whence, in this numerical example:
Re=20 ohms.
The current Ie, for an overvoltage V1 of 18 volts, is equal in this example to 300 milliamperes.

Generally, the limiter has a threshold E ranging from 10 to 15 volts (E>Vmax, E<V1) and a low dynamic resistance Re (up to some tens of ohms) so as to let through a great deal of current: a high gain in latch-up current is sought.

In a first exemplary embodiment, illustrated in FIG. 3, a limiter is formed by zener diodes in series, such that the sum of the threshold voltages of the diodes is equal to the triggering voltage E of the limiter. Alternatively, it is possible to use a single diode with the desired threshold voltage or, more generally, a series association of components that can be likened, in operation, to diodes in reverse (diode-mounted transistor for example).

In a preferred example (FIG. 3), there will be two zener diodes in series, of the CMOS type, each formed by a P+/N+ junction (not shown) in a well, with a typical threshold voltage of the order of 6 volts. A practical exemplary embodiment of such diodes is described in the Italian patent No. 2 2228 A/89 filed on behalf of Sgs-Thomson Microelectronics s.r.l. on 31st October 1989, as well as corresponding U.S. application Ser. No. 07/604,895 and corresponding European application EP90/2,028,406, all of which are hereby incorporated by reference.

This diode device is equivalent to a single limiter likened to a Thevenin generator with E, Re characteristics of the order of 12 volts and 20 ohms. However, it does not intrinsically withstand electrostatic discharges, another phenomenon to which the CMOS circuits are highly sensitive. Indeed, the electrostatic discharges may induce currents of several amperes. A protection device against the electrostatic discharges therefore has to be added on to the limiter. The CMOS internal circuitry 1 is itself already protected in a standard way from electrostatic discharges.

In a known way (FIG. 2) the protection device against electrostatic discharges is a lateral bipolar transistor T, for example of the NPN type in a P− substrate. It is also possible to use a PNP type lateral transistor made in an N type well. In the example, the NPN type lateral bipolar transistor is obtained in a P− type substrate with the following from left to right: a first N+ type diffusion c connected to Vcc, a second N+ type diffusion e connected to Vss and a third P+ type substrate contact diffusion connected to Vss. The diffusion c is the collector and the diffusion e is the emitter of the NPN type protection bipolar transistor T in this example. Its base b is constituted by the substrate zone between the emitter and the collector, and biased through a lateral resistor R1 between the P− substrate and the P+ diffusion zone. The diffusions are insulated from one another by a field oxide layer 4 and, beneath each field oxide layer, a zone 5 with higher doping than the P− substrate, this zone 5 being generally called a P−iso zone or, again, a "field implantation" zone. The triggering threshold of the transistor T is linked to the avalanche voltage of its collector-base junction. This voltage Vo is linked to the surface doping in this zone of the substrate and, more specifically, by the P-iso zone 5 between the two N+ type zones e and c. In the case of typical, present-day CMOS methods, this voltage Vo is of the order of 20 volts.

Now, the limiter triggering voltage E is of the order of 12 volts and is always lower than 20 volts. The threshold E is therefore lower than that of the device T for protection against electrostatic discharges and, should there be electrostatic discharges, then only the

limiter will be on. This protection device 4 described in FIG. 2 therefore has no purpose and the limiter risks being destroyed.

Thus, a protection resistor R'e is interposed in series between the terminal A (Vcc) and the limiter (FIG. 3). Between 12 and 20 volts, this resistor R'e enables the voltage at the terminals of the protection device 4 to reach at least 20 volts, thus enabling it to be triggered and hence enabling the discharge current I'e in the limiter to be limited.

The maximum current allowable in the limiter is referenced Icrit. In practice, the current I'e going through the limiter should be lower than Icrit, i.e.: I'e<Icrit Icrit, Vo, E, Re being known and I'e in FIG. 3 being expressed by:

$$I'e = \frac{Vo - E}{Re + R'e}$$

I'e<Icrit leads to:

$$R'e > \frac{Vo - E}{Icrit} - Re$$

It is furthermore sought to have a maximum gain in current in the limiter for protection against latch-up: it is therefore necessary to choose R'e at a value that is as low as possible. In practice, it is possible to approach the minimum value Rmin given in the above inequality, namely:

$$Rmin = \frac{Vo - E}{Icrit} - Re$$

Preferably, for greater security, the maximum current flowing through the limiter will be chosen rather so that it is Icrit/2, and hence:

$$R'e = \frac{2\,(Vo - E)}{Icrit} - Re \qquad (1)$$

will be chosen.

The current flowing in the limiter in the case of overvoltage V1 (applied between the terminals A and B) is then:

$$I'e = \frac{(V1 - E)}{Re + R'e} \qquad (2)$$

In a numerical example, with Icrit=400 mA, Vo=20 volts, Re=20 ohms, V1=18 volts and E=12 volts, from (1) it follows that: R'e=20 Ohms and from (2) it follows that: I'e=150 mA.

The apparent current for the triggering of the latch-up of the CMOS circuit is therefore:

$$I'1 = I1 + I'e = 20 + 150 = 170\ mA$$

Thus, the gain in current is smaller than it is without protection against electrostatic discharges (there is a change from I'1=300 mA to I'1=170 mA), but its level remains very satisfactory.

Another exemplary embodiment of the limiter (FIG. 4) uses a lateral bipolar transistor T' like the one seen here above for the protection device against electrostatic discharges (FIG. 2), but made by overdoping its base b, i.e. by overdoping the P-iso zone 6 (FIG. 4) between the two N+ zones which form the emitter e' and the collector c'. It has indeed been seen that, in doing so, an adjustment is made in the threshold Vo for the triggering of this standard protection device against electrostatic discharges (by modification of the collector-base avalanche voltage). Thus, it is possible to set the threshold Vo for the triggering of the transistor T' at a value between 10 and 15 volts, for example at 12 volts. In such a device, when it is triggered, the dynamic resistance is almost zero: it therefore absorbs all the current without limit. It therefore becomes impossible to trigger the latch-up: we have a limiter (FIG. 5) with characteristics of E=12 volts and Re=0 ohms.

Thus, by having its threshold lowered, the protection device against electrostatic discharges has also become a protection device against latch-up. However, this device is costlier than in the first embodiment, for it requires an additional masking/implantation level in the fabrication of the CMOS circuit, to overdope the base (the making of the zone 6). By contrast, its efficiency is total, because it radically bars the latch-up and because it provides intrinsic protection against electrostatic discharges.

Other embodiments of limiters with the requisite conditions (E between 10 and 15 volts approximately, Re low) are naturally possible to provide protection against latch-up according to the invention.

What is claimed is:

1. A CMOS circuit protected against latch-up, comprising:

internal circuitry connected between a first more positive supply terminal and a second more negative supply terminal to draw power therefrom, said internal circuitry having, as known parameters, a current I_1 and a voltage V_1 for the triggering of the latch-up phenomenon;

a limiter, connected, in parallel with the internal circuitry, between the first supply terminal and the second supply terminal, this limiter having a triggering threshold below voltage V_1 and consuming current of a value far higher than I_1 when a voltage of V_1 or greater is applied between the first and second supply terminals.

2. A CMOS circuit according to claim 1, wherein the triggering threshold of said limiter is higher than a maximum voltage permitted for the operation of the CMOS internal circuitry.

3. A CMOS circuit according to claim 1, wherein said limiter comprises zener diodes that are operatively connected in series between the first supply terminal and the second supply terminal, the sum of the reverse-breakdown threshold voltages of the diodes being equal to the triggering threshold of said limiter.

4. A CMOS circuit according to claim 1, wherein said limiter comprises two zener diodes in series, each diode having a threshold of the order of 6 volts.

5. A CMOS circuit according to claim 1, further comprising a protection circuit against electrostatic discharges which is connected, in parallel with said limiter, between said first supply terminal and said second supply terminal.

6. A CMOS circuit according to claim 1, further comprising a protection resistor which is placed in series between said first supply terminal and said limiter.

7. A CMOS circuit according to claim 1, further comprising: a protection resistor which is placed in series between said first supply terminal and said limiter; and a lateral bipolar transistor, having a collector con-

nected to said first supply terminal, and an emitter and a base connected to said second supply terminal.

8. A CMOS circuit according to claim 1, wherein said limiter is a lateral bipolar transistor, one collector of which is connected to said first supply terminal, and one emitter and one base of which is connected to said second supply terminal, and wherein said base has been overdoped to diminish a collector-base avalanche voltage.

9. The integrated circuit of claim 1, wherein said limiter has a threshold voltage in the range of about 10–15 V inclusive.

10. The integrated circuit of claim 1, wherein said power supply connections are specified for connection for approximately 5 Volts of supply voltage, and said limiter has a threshold voltage of about 12 V.

11. The integrated circuit of claim 1, wherein said limiter has a dynamic on-resistance of no more than a few tens of ohms.

12. The integrated circuit of claim 1, wherein said limiter comprises multiple Zener diodes in series.

13. A CMOS integrated circuit, comprising:

first and second power supply input connections;

a plurality of active devices connected to implement a desired circuit configuration, and operatively connected to be powered from said first and second power supply input connections, and configured to reliably withstand peak voltages of at least a specified value Vmax;

a limiter circuit, having a nonlinear response with a low dynamic impedance R_d at applied voltages greater than a threshold voltage E; said limiter being connected, in series with a resistance, between said first and second power supply connections;

an electrostatic discharge protection device, connected directly between said first and second power supply connections, said electrostatic discharge protection device having a nonlinear response with a very low impedance at applied voltages greater than a threshold voltage Vo;

wherein said limiter and said electrostatic discharge protection device have respective device parameters such that $Vmax<E<Vo$.

14. The integrated circuit of claim 13, wherein said limiter has a threshold voltage in the range of about 10–15 V inclusive.

15. The integrated circuit of claim 13, wherein said power supply connections are specified for connection to approximately 5 Volts of supply voltage, and said limiter has a threshold voltage of about 12 V.

16. The integrated circuit of claim 13, wherein said limiter has a dynamic on-resistance of no more than a few tens of ohms.

17. The integrated circuit of claim 13, wherein said limiter has a dynamic on-resistance of about 20 ohms.

18. The integrated circuit of claim 13, wherein said limiter comprises multiple Zener diodes in series.

19. A CMOS integrated circuit, comprising:

first and second power supply input connections;

a plurality of active devices connected to implement a desired circuit configuration, and operatively connected to be powered from said first and second power supply input connections, and configured to reliably withstand peak voltages of at least a specified value Vmax; said plurality of active devices collectively having a latchup trigger voltage of least V1, and a latchup holding current of at least I1;

a limiter circuit, comprising a Zener diode structure having a threshold voltage E; said limiter being connected, in series with a resistance, between said first and second power supply connections;

an electrostatic discharge protection device, comprising a diode-connected bipolar transistor connected directly between said first and second power supply connections, said electrostatic discharge protection device having a threshold voltage Vo;

wherein said limiter and said electrostatic discharge protection device have respective device parameters such that $Vmax<E$, $E<V1$, and $E<Vo$.

20. The integrated circuit of claim 19, wherein said limiter has a threshold voltage in the range of about 10–15 V inclusive.

21. The integrated circuit of claim 19, wherein said power supply connections are specified for connection to approximately 5 Volts of supply voltage, and said limiter has a threshold voltage of about 12 V.

22. The integrated circuit of claim 19, wherein said limiter has a dynamic on-resistance of no more than a few tens of ohms.

23. The integrated circuit of claim 19, wherein said limiter has a dynamic on-resistance of about 20 ohms.

24. The integrated circuit of claim 19, wherein said limiter comprises multiple Zener diodes in series.

* * * * *

US005515225A

United States Patent [19]

Gens et al.

[11] Patent Number: 5,515,225

[45] Date of Patent: May 7, 1996

[54] **INTEGRATED CIRCUIT PROTECTED AGAINST ELECTROSTATIC OVERVOLTAGES**

[75] Inventors: **Marc Gens**, Saint Martin D'Uriage; **Jean Jimenez**, Voiron, both of France

[73] Assignee: **SGS-Thomson Microelectronics S.A.**, Saint-Genis, Pouilly, France

[21] Appl. No.: **286,490**

[22] Filed: **Aug. 4, 1994**

[30] **Foreign Application Priority Data**

Aug. 6, 1993 [FR] France 93 09928

[51] **Int. Cl.**[6] **H02H 9/04**

[52] **U.S. Cl.** **361/56**; 361/91; 361/111

[58] **Field of Search** 361/56, 91, 118, 361/90, 93, 100–101, 111, 212

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,819,047	4/1989	Gilfeather et al.	357/23.13
4,996,626	2/1991	Say	361/91
5,034,845	7/1991	Murakami .	
5,239,440	8/1993	Merrill .	

FOREIGN PATENT DOCUMENTS

0148577 7/1985 European Pat. Off. H02H 9/04

OTHER PUBLICATIONS

Patent Abstracts Of Japan, vol. 9, No. 53 (E–301)(1776) Mar. 7, 1985 & JP–A–59191371.

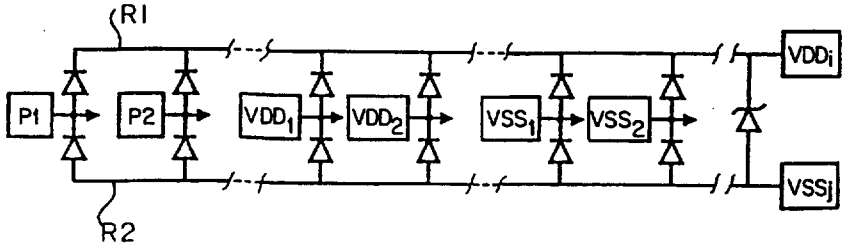
Revue Generale De L'Electricite, No. 2, Feb. 1987, Paris, FR, pp. 19–24, C. Chapron, "Les Circuits Integres Bipolaires: Comment Diminuer Leur Sensibilite Aux Decharges Electrostatiques".

Primary Examiner—A. D. Pellinen
Assistant Examiner—Michael J. Sherry
Attorney, Agent, or Firm—David M. Driscoll; James H. Morris

[57] **ABSTRACT**

A circuit protects pads of an integrated circuit having a plurality of power supply sources against electrostatic overvoltages. Each power supply is connected between a high voltage pad and a low voltage pad. Each pad of the circuit, as well as each power supply pad, is connected to the anode of a first diode having its cathode connected to a first conductive bus, and to the cathode of a second diode having its anode connected to a second conductive bus. A unidirectional clipping device is connected by its cathode to the first bus and by its anode to the second bus.

23 Claims, 2 Drawing Sheets



Ex C

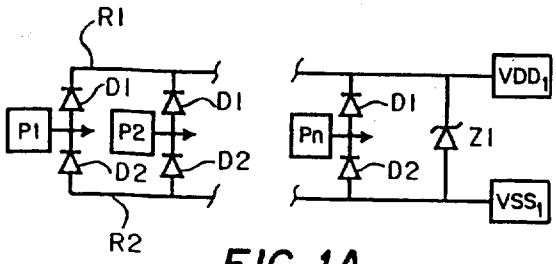


FIG. 1A
(PRIOR ART)

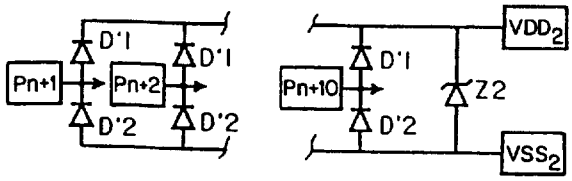


FIG. 1B
(PRIOR ART)

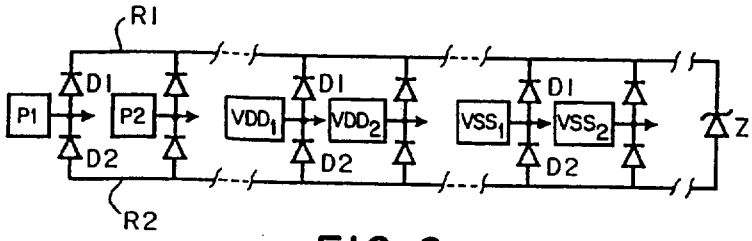


FIG. 2

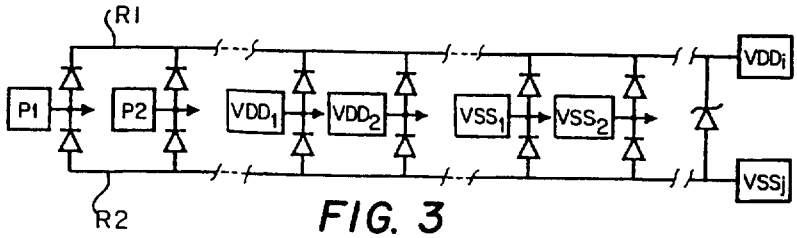


FIG. 3

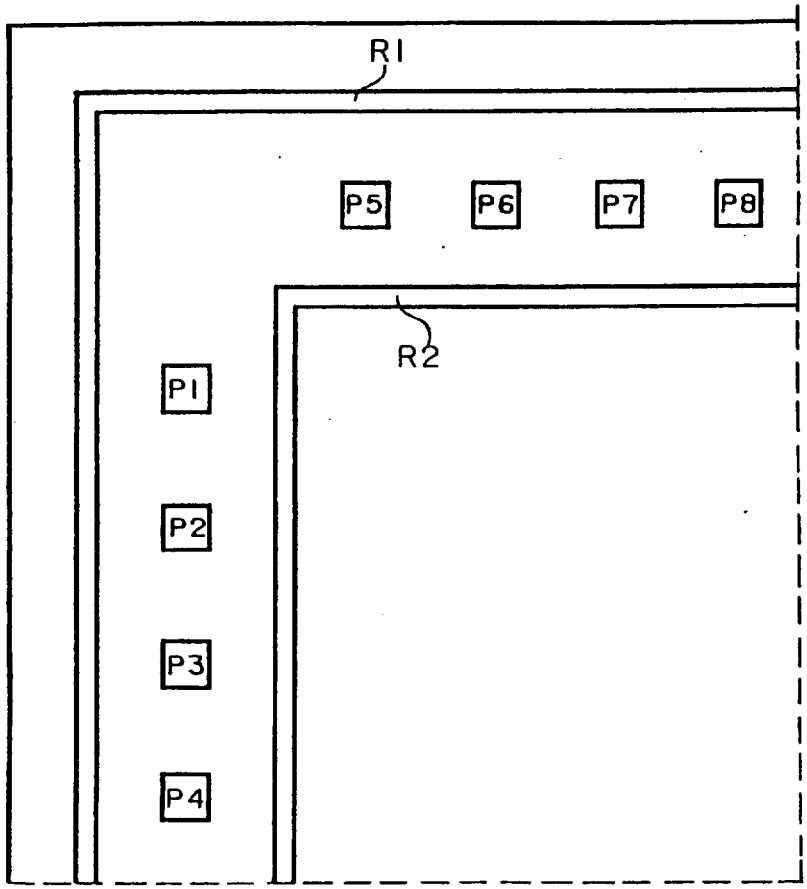


FIG. 4

INTEGRATED CIRCUIT PROTECTED AGAINST ELECTROSTATIC OVERVOLTAGES

BACKGROUND OF THE INVENTION

1. Field of the Invention

Integrated circuits include various components that can be destroyed by the occurrence of overvoltages. A particular problem is the protection against electrostatic overvoltages of the terminals of integrated circuits during their final manufacturing steps or handling before mounting. These electrostatic overvoltages are, for example, caused when two pads or pins of an integrated circuit are contacted by a handling tool or the fingers of a user.

2. Discussion of the Related Art

It should be noted that protection circuits against electrostatic overvoltages must be active even if the integrated circuit is not connected and, more particularly, even if its ground terminals are not interconnected. In addition, these protection circuits against electrostatic pulses must not impair the operation of the integrated circuit once it is connected.

FIG. 1A shows a conventional protection scheme of an integrated circuit against electrostatic pulses including a single power supply source connected between a high voltage pad VVD_1 and a low voltage pad VSS_1. Each input/output pad p1-pn of the circuit is connected to a supply bus R1 connected to the high voltage pad VDD_1 through an insulating diode D1 having its anode connected to the pad and its cathode connected to the supply bus R1. Each pad P1-Pn is also connected to a supply bus R2 connected to pad VSS_1 through an insulating diode D2 having its cathode connected to the pad and its anode connected to the supply bus R2. The supply bus R1 is connected to the bus R2 through a clipping device symbolized in the form of an avalanche diode Z1 having its cathode connected to the bus R1 and its anode connected to the bus R2. Any conventional unidirectional clipping system can be used as a clipping device Z1. The clipping device Z1 has an avalanche threshold voltage higher than VDD_1- VSS_1 but, of course, lower than the value of the overvoltage against which the components of the integrated circuit are to be protected.

In practice, in integrated circuits, diodes D1, D2 and the clipping device Z1 are disposed inside the integrated circuit and buses R1 and R2 are metallizations formed on this integrated circuit. It should be noted that the surface area off the components corresponding to diodes D1 and D2 is relatively small, in contrast, to a relatively large surface area of the clipping device Z1 of the integrated circuit.

The circuit of FIG. 1A meets the requisites for protection against electrostatic pulses when the integrated circuit has a single external power supply source.

However, a drawback of the circuit of FIG. 1A is that, since each pad is connected to the high voltage supply through a forward biased diode and to the low voltage supply through a reverse biased diode, each pad cannot exceed 0.6 volt above the high voltage VDD or decrease down to −0.6 volt below the low voltage VSS (usually ground). In practice, in some cases, it is desirable that the pad voltage can exceed the highest supply voltage. For example, when an integrated circuit is in operation, if the high supply voltage is interrupted and grounded and if one of the pads is associated with a storing capacitor, this capacitor will be discharged through one of diodes D1 toward the power supply pad VDD_1. Also, output amplifiers

can be connected to inductive loads. When overvoltages occur, they are clipped by the connection toward the high power supply terminal, which may be undesirable.

Existing integrated circuits often include a plurality of power supply sources (up to 7 in some circuits). Each of these power supply sources includes a high power supply terminal VDD_i and a low power supply terminal VSS_i.

Conventionally, to protect a circuit having a plurality of power supply sources, the circuit of FIG. 1A is simply enlarged by duplicating it by circuits such as the circuit of FIG. 1B. Pads P1-Pn, that are part of circuit portions connected to a first power supply source VDD_1-VSS_1, are connected by the circuit of FIG. 1A. Pads Pn+1-Pn+k, that are part of circuit portions connected to a second power supply source VDD_2-VSS_2, are connected by a circuit analogous to the circuit of FIG. 1A, illustrated in FIG. 1B. This type of circuit is duplicated for each power supply source.

The use of this type of integrated circuit protection including a plurality of power supply sources has several major drawbacks.

A first drawback is that such a structure requires as many clipping devices Zi as power supply sources. However, as indicated above, these clipping devices occupy a relatively large integrated circuit surface area, which conflicts with the integrated circuit designer's desire to decrease the surface area of the integrated circuit.

A second, still more important, drawback of the simultaneous use of circuits such as those of FIGS. 1A and 1B in a same integrated circuit is that this type of protection is ineffective when an electrostatic pulse is generated between a pad P1-Pn associated with a first power supply source and a pad P_{n+1}-P_{n+k} associated with a second power supply source. Then, current no longer flows through a clipping device to short the overload, which then propagates inside the integrated circuit and can be destroying.

SUMMARY OF THE INVENTION

Thus, an object of the present invention is to provide, for an integrated circuit with a multiplicity of power supply sources, a protection circuit that is fully effective against electrostatic pulses occurring between pads or pins, when the integrated circuit is not connected.

Another object of the invention is to provide such a protection circuit wherein the voltage of the pads can increase to a voltage higher than the highest supply voltage or decrease below the lowest supply voltage.

A further object of the present invention is to provide a protection circuit that occupies a reduced surface area.

These objects are achieved according to the invention with a circuit for protecting pads of an integrated circuit, associated with a plurality of power supply sources, against electrostatic overvoltages. Each power supply source is connected to a high voltage pad and to a low voltage pad. In this circuit, each pad of the circuit, as well as power supply pads, is connected to the anode of a first diode having its cathode connected to a first conductive bus, and to the cathode of a second diode having its anode connected to a second conductive bus; and a unidirectional clipping device is connected by its cathode to the first bus and by its anode to the second bus.

According to an embodiment of the invention, all the supply pads are connected in the way indicated above.

According to an embodiment of the invention, the high voltage pad of the highest power supply source is connected

to the first bus, and the low voltage pad of the lowest power supply source is connected to the second bus.

According to an embodiment of this invention, the buses are comprised of conductive paths formed at the periphery of the integrated circuit, on the inside and on the outside of the connecting pads, respectively.

The foregoing and other objects, features, aspects and advantages of the invention will become apparent from the following detailed description of the present invention when taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF DRAWINGS

FIG. 1A represents a conventional electrostatic protection circuit for protecting pads of an integrated circuit associated with one supply source;

FIGS. 1A and 1B, taken in conjunction, represent a conventional electrostatic protection circuit for protecting pads of integrated circuits associated with two supply sources;

FIG. 2 represents an electrostatic protection circuit for protecting integrated circuit pads according to the invention;

FIG. 3 represents an alternative electrostatic protection circuit for protecting integrated circuit pads according to the invention; and

FIG. 4 schematically represents a portion of a chip of an integrated circuit incorporating an electrostatic protection circuit according to the invention.

DETAILED DESCRIPTION

FIG. 2 schematically represents an electrostatic overvoltage protection circuit according to the invention for protecting an integrated circuit associated with a plurality off power supply sources. Each power supply source is applied between a high supply terminal VDD_1, VDD_2, ... and a low supply terminal VSS_1, VSS_2, ... The integrated circuit also includes conventional input/output pads P1, P2, . . . The protection circuit according to the invention includes buses R1 and R2 that are respectively connected to the cathode and to the anode of a clipping device Z. Each pad P1, P2, . . . is connected to bus R1 through a forward biased diode D1 and to bus R2 through a reverse biased diode D2. In addition, according to the invention, each of the high and low power supply pads, VDD_1, VDD_2, . . . , VSS_1, VSS_2, . . . , respectively, is also connected to buses R1 and R2 through forward and reverse biased diodes D1 and D2, respectively. In the embodiment of FIG. 2, buses R1 and R2 are held floating.

Thus, according to the invention, an electrostatic pulse between any one of pads P1, P2, . . . , and any other of these pads or any one of the high or flow power supply pads, will pass through a diode D1, the clipping device Z, and a diode D2.

Accordingly, this type of protection ensures full protection against electrostatic pulses between any pair of pads of the integrated circuit.

An advantage of the protection circuit according to the invention is that it includes a single clipping device Z disregarding the number of power supplies of the circuit. This provides a decrease in the surface area of the integrated circuit.

A further advantage of such a circuit is that the voltage of any one of pads P1, P2, . . . , can, during the operation of the integrated circuit, increase to a voltage higher than the voltage of the highest power supply terminal or decrease to a voltage lower than the voltage of the lowest power supply terminal. Indeed, the voltage difference between two pads is constantly limited by the clipping voltage of the clipping device Z (plus two forward voltage drops of a diode).

FIG. 3 illustrates an alternative embodiment of the invention showing substantially the same circuit as the circuit of FIG. 2 but wherein the pad to be connected to the highest supply voltage VDD_i is connected to bus R1, and wherein the pad to be connected to the lowest supply voltage VSS_j is connected to bus R2. This alternative embodiment also ensures protection against all the types of overvoltage, and allows, with respect to the above embodiment, to suppress four insulating diodes that would be associated, in the embodiment of FIG. 2, with pads VDD_i and VSS_j. However, the drawback of this alternative embodiment is to prevent one of the pads P1, P2, . . . , from substantially exceeding the highest voltage or from substantially decreasing below the lowest voltage.

FIG. 4 is a schematic top view off an integrated circuit portion representing pads P1–P8 that can be input/output pads or power supply pads. Conventionally, these pads are disposed at the periphery of the integrated circuit. Preferably, buses R1 and R2 are also disposed at the periphery of the integrated circuit chip, in the vicinity of the pads, on the inside and on the outside of the pads, respectively, with respect to the middle of the chip. The pads are connected to buses R1 and R2 through diodes that are conventionally incorporated in the circuit.

As is apparent to those skilled in the art, various modifications can be made to the above disclosed preferred embodiments. For example, in the case of an integrated circuit associated with a multiplicity of power supply sources in which the various sources are insulated one from the other, while having the same value, each high voltage supply pad can be coupled to the bus that is connected to the cathode of the protection circuit through a pair of parallel, head-to-tail diodes, and each low voltage pad can be coupled to the bus that is connected to the anode of the clipping device through a pair of head-to-tail diodes.

Having thus described one particular embodiment of the invention, various alterations, modifications, and improvements will readily occur to those skilled in the art. Such alterations, modifications, and improvements are intended to be part of this disclosure, and are intended to be within the spirit and scope of the invention. Accordingly, the foregoing description is by way of example only and is not intended to be limiting. The invention is limited only as defined in the following claims and the equivalents thereto.

What is claimed is:

1. A circuit for protecting a plurality of input/output pads of an integrated circuit associated with a plurality of power supply sources against electrostatic overvoltages, each power supply source being connected to a high voltage pad and a low voltage pad, comprising:

a first conductive bus and a second conductive bus;

a plurality of first diodes coupled to the first conductive bus;

a plurality of second diodes coupled to the second conductive bus, wherein each of the plurality of input/output pads, at least one high voltage pad, and at least one low voltage pad is connected to an anode of a respective first diode having a cathode connected to the first conductive bus, and to a cathode of a respective second diode having an anode connected to the second conductive bus; and

a unidirectional clipping device having an anode connected to the second conductive bus, and a cathode connected to the first conductive bus.

2. The protection circuit of claim 1, wherein the high and the low voltage pads connected to each of the plurality of power supply sources, are connected to an anode of a respective first diode having a cathode connected to the first conductive bus, and connected to a cathode of a respective second diode having an anode connected to the second conductive bus.

3. The protection circuit of claim 1, wherein a high voltage pad of a highest power supply source is connected to the first conductive bus, and a low voltage pad of a lowest power supply source is connected to the second conductive bus.

4. The protection circuit of claim 1, wherein each of the first and the second conductive buses is formed at a periphery of the integrated circuit, the input/output and voltage pads being disposed between the first and the second conductive buses.

5. A protection circuit for protecting an integrated circuit having a plurality of pads including a first circuit pad, a second circuit pad, a first voltage supply pad that connects to a first voltage supply and a second voltage supply pad that connects to the first voltage supply, the protection circuit comprising:

a first conductive path and a second conductive path;

a plurality of first diodes, a first diode being coupled between each pad and the first conductive path;

a plurality of second diodes, a second diode being coupled between each pad and the second conductive path; and

a clipping circuit coupled between the first and the second conductive paths.

6. A protection circuit for protecting an integrated circuit having a plurality of pads including a first circuit pad, a second circuit pad, a first voltage supply pad that connects to a first voltage supply, and a second voltage supply pad that connects to the first voltage supply, the protection circuit comprising:

a first conductive path and a second conductive path;

a plurality of first diodes, a first diode being coupled between each pad and the first conductive path;

a plurality of second diodes, a second diode being coupled between each pad and the second conductive path; and

a clipping circuit coupled between the first and the second conductive paths, wherein

each diode has an anode and a cathode,

each pad is coupled to the anode of the first diode that is coupled between the pad and the first conductive path;

each pad is coupled to the cathode of the second diode that is coupled between the pad and the second conductive path;

the first conductive path is coupled to the cathode of each first diode; and

the second conductive path is coupled to the anode of each second diode.

7. The protection circuit of claim 6, wherein the clipping circuit includes a unidirectional clipping device having an anode and a cathode, the anode being coupled to the second conductive path, and the cathode being coupled to the first conductive path.

8. The protection circuit of claim 7, wherein the unidirectional clipping device includes a zener diode having a clipping voltage.

9. The protection circuit of claim 8, wherein each first diode has a first forward voltage drop, each second diode has a second forward voltage drop, and the protection circuit has a voltage protection threshold equal to approximately the sum of the first forward voltage drop, the second forward voltage drop and the clipping voltage.

10. The protection circuit of claim 7, wherein the first and the second conductive paths include a first terminal that connects to a second voltage supply, and a second terminal that connects to a third voltage supply, respectively to provide a first potential difference between the first and the second terminals, which is greater than a second potential difference between the first voltage supply pad and the second voltage supply pad.

11. The protection circuit of claim 7, wherein the pads and the conductive paths are disposed adjacent a periphery of the integrated circuit, and the pads are disposed between the first and the second conductive paths.

12. A protection circuit for protecting an integrated circuit having a plurality of external connecting means for establishing an external connection including first circuit means for establishing an external circuit connection, second circuit means for establishing an external circuit connection, first voltage supply means for establishing first external voltage connection to a first voltage supply and second voltage supply means for establishing a second external voltage connection to the first voltage supply, the protection circuit comprising:

first conducting means for conducting current and second conducting means for conducting current;

a plurality of first voltage controlling means for controlling voltage, a first voltage controlling means being coupled between each external connecting means and the first conducting means;

a plurality of second voltage controlling means for controlling voltage, a second voltage controlling means being coupled between each external connecting means and the second conducting means; and

clipping means for clipping voltage coupled between the first and a second conducting means.

13. The protection circuit of claim 12, wherein

each voltage controlling means has input means for inputting current and output means for outputting current,

each external connecting means is coupled to the input means of the first voltage controlling means that is coupled between the external connecting means and the first conducting means;

each external connecting means is coupled to the output means of the second voltage controlling means that is coupled between the external connecting means and the second conducting means;

the first conducting means is coupled to each output means of each first voltage controlling means; and

the second conducting means is coupled to each input means of each second voltage controlling means.

14. The protection circuit of claim 13, wherein the clipping means includes clipping input means and clipping output means, the clipping input means of the clipping means being coupled to the second conducting means, and the clipping output means of the clipping means being coupled to the first conducting means.

15. The protection circuit of claim 14, wherein the clipping means includes means for conducting current in a reverse direction so that current enters through the clipping output means and exits through the clipping input means when a voltage between the first conducting means and the second conducting means exceeds a clipping voltage.

16. The protection circuit of claim 15, wherein each first voltage controlling means has a first forward voltage drop,

each second voltage controlling means has a second forward voltage drop, and the protection circuit has a voltage protection threshold equal to approximately the sum of the first forward voltage drop, the second forward voltage drop and the clipping voltage.

17. The protection circuit of claim 14, wherein the first and the second conducting means include a first terminal that connects to a second voltage supply, and a second terminal that connects to a third voltage supply, respectively, to provide a first potential difference between the first and the second terminals, which is greater than a second potential difference between the first and the second voltage supply means.

18. The protection circuit of claim 14, wherein the external connecting means and the conducting means are disposed adjacent a periphery of the integrated circuit, and the external connecting means are disposed between the first and the second conducting means.

19. A protection method for protecting an integrated circuit having a first circuit pad, a first voltage supply pad that connects to a first voltage supply, and a second voltage supply pad that connects to the first voltage supply, the method including:

draining current from the first circuit pad through a first diode, a clipping device having a clipping voltage, and a second diode, each diode having a forward bias voltage, if a potential difference between the first circuit pad and the first voltage supply pad exceeds a sum of the forward bias voltages of the first and second diodes, and the clipping voltage; and

providing current to the first circuit pad through third diode, the clipping device, and a fourth diode if a potential difference between the first circuit pad and the second voltage supply pad exceeds the sum.

20. The method of claim 19, wherein the integrated circuit includes a second pad, and the method includes the step of:

draining current from the first pad and providing current to the second pad when a potential difference between the first and second circuit pads exceeds the sum.

21. The protection method of claim 19, further including the steps of:

connecting a positive terminal of a first voltage supply to the first voltage pad; and

connecting a negative terminal of the first voltage supply to the second voltage pad.

22. The protection method of claim 21, further including the step of:

draining current from the first circuit pad, through the first diode, to a third voltage supply.

23. The protection method of claim 21, further including the step of:

providing current to the first circuit pad, through the second diode, from a third voltage supply.

* * * * *

US005946261A

United States Patent [19]

Artieri

[11] Patent Number: 5,946,261

[45] Date of Patent: *Aug. 31, 1999

[54] **DUAL-PORT MEMORY**

[75] Inventor: Alain Artieri, Meylan, France

[73] Assignee: SGS-Thomson Microelectronics S.A., Saint Genis, France

[*] Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

[21] Appl. No.: **08/643,735**

[22] Filed: **May 6, 1996**

Related U.S. Application Data

[63] Continuation of application No. 08/273,047, Jul. 8, 1994, abandoned.

[30] **Foreign Application Priority Data**

Jul. 12, 1993 [FR] France 93-08837

[51] **Int. Cl.**[6] **G11C 7/00**

[52] **U.S. Cl.** **365/230.05**; 365/203; 365/210

[58] **Field of Search** 365/189.04, 189.07, 365/189.12, 210, 240, 49, 230.05, 203

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,849,935	7/1989	Miyazawa	365/89.05
4,864,543	9/1989	Ward et al.	365/221
5,245,575	9/1993	Sasaki et al.	365/189.12
5,268,869	12/1993	Ferris et al.	365/210
5,317,537	5/1994	Shinagawa et al.	365/210 X
5,424,995	6/1995	Miyazaki et al.	365/230.05

FOREIGN PATENT DOCUMENTS

0 390 453 A2 10/1990 European Pat. Off. .

OTHER PUBLICATIONS

Proceedings of the IEEE 1989 Custom Integrated Circuits Conference, May 15, 1989, San Diego, CA, pp. 23.4.1–23.4.4. Kawauchi, et al., "A 1.0um Compilable FIFO Buffer for Standard Cell".

1990 Symposium on VLSI Circuits Digest of Technical Papers, Jun. 7, 1990, Honolulu, HI, pp. 65–66. Sakaue, et al., "A 0.8um BiCMOS ATM Switch on the 800Mbps asynchronous buffered banyan network".

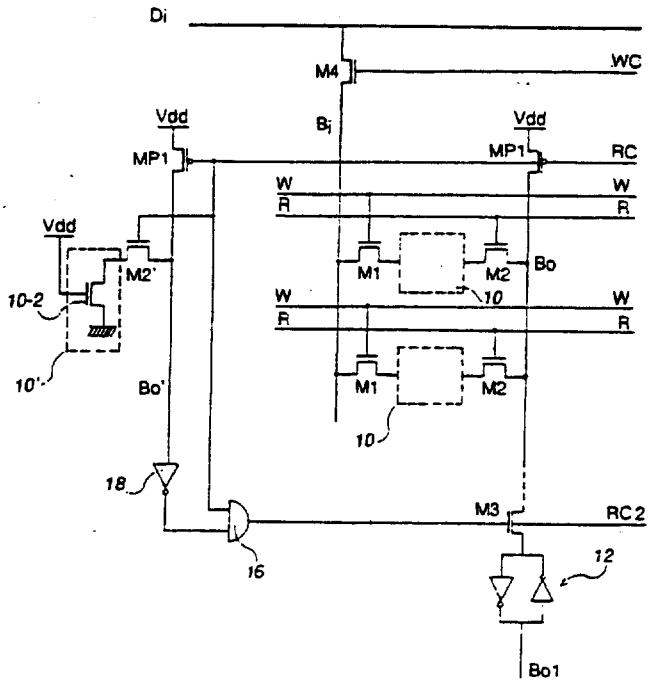
Primary Examiner—Do Hyun Yoo

Attorney, Agent, or Firm—Wolf, Greenfield & Sacks, P.C.

[57] **ABSTRACT**

A dual-port memory includes a dummy memory cell associated with a dummy output line and with a precharge transistor, the output of the dummy cell being at "0". A dummy read transistor is turned on by the active state of the read selection signal and connects the output of the dummy cell to the dummy output line. Circuitry is provided for turning on the output transistors of the memory when the state of the dummy output line reaches a predetermined switching threshold of an inverter.

7 Claims, 4 Drawing Sheets



EX D

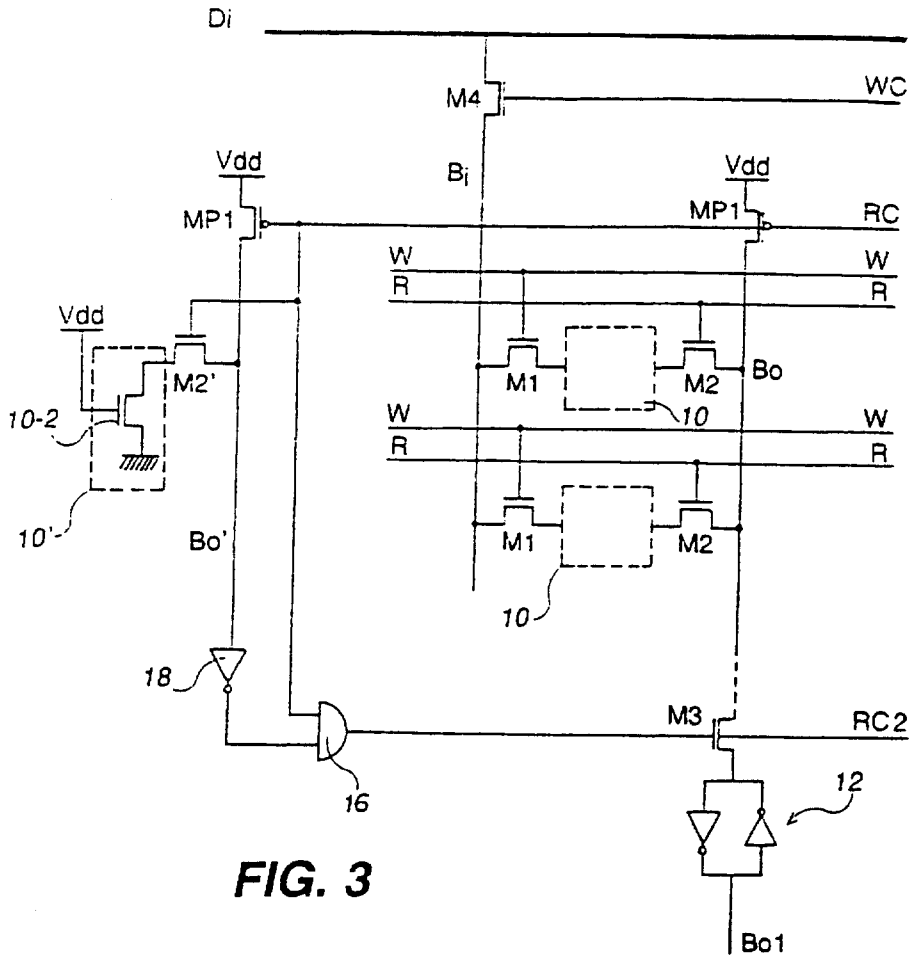

Di
M4
WC
Vdd
Bi
Vdd
MP1
MP1
RC
W
R
W
R
Vdd
M2'
M1
M2
Bo
10-2
10
W
R
W
R
10'
Bo'
M1
M2
10
18
M3
RC2
16
12
Bo1

FIG. 3

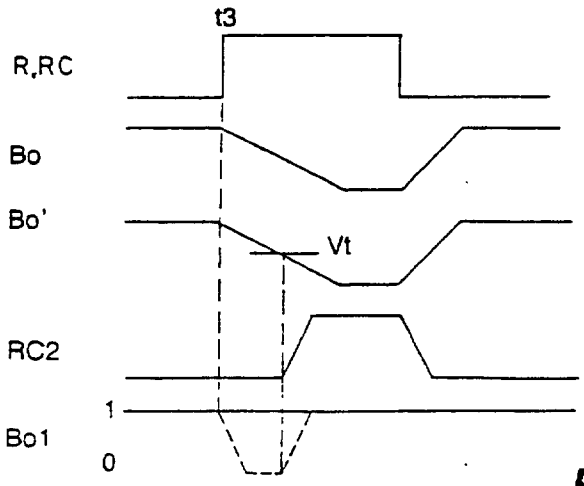

t3
R,RC
Bo
Bo'
Vt
RC2
1
Bo1
0

FIG. 4

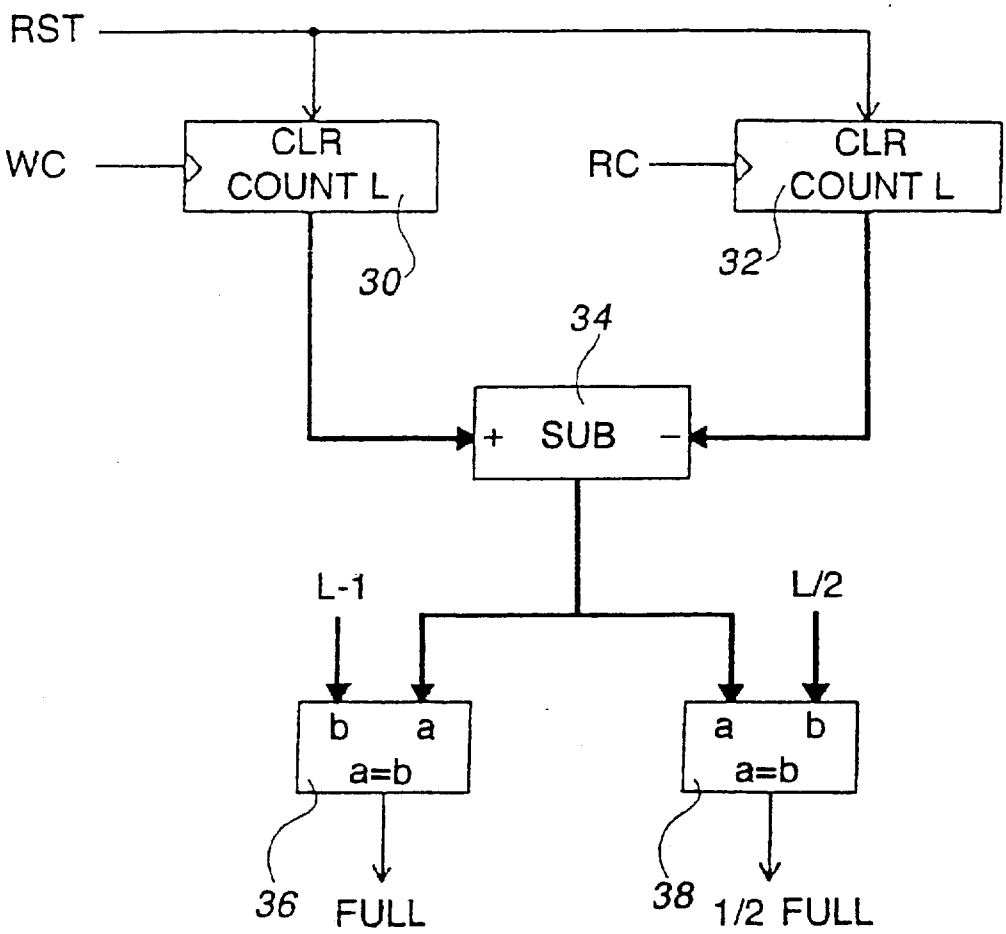

RST
WC
CLR
COUNT L
RC
CLR
COUNT L
30
32
34
+ SUB −
L-1
L/2
b a
a=b
a b
a=b
36
FULL
38
1/2 FULL

FIG. 6

These and other objects are achieved with a dual-port memory including dual-port memory cells disposed in columns. One output line per column is connected to each cell of the column through a read transistor that is controlled by a respective read line. Precharge transistors connect the output lines to a precharge voltage corresponding to a first logic state, these transistors being turned on by the inactive state of a read selection signal. Output transistors connect the output lines to a memory output. A dummy memory cell is associated with a dummy output line and with a precharge transistor; the output of this dummy cell is at a second logic state opposite to the first logic state. A dummy read transistor is turned on by the active state of the read selection signal and connects the output of the dummy cell to the dummy output line. Means are provided for turning on the output transistors when the state of the dummy output line reaches a predetermined threshold between the first and second logic states.

According to an embodiment of the invention, one input line per column is connected to each cell of the column through a write transistor controlled by a respective write line. For each write line, the memory includes a transistor that connects this write line to a memory input and is turned on by the active state of a write selection signal.

According to an embodiment of the invention, the means for turning on the output transistors include a logic gate having a first input connected to the dummy output line through an inverter and a second input that receives the read selection signal.

According to an embodiment of the invention, the dummy cell includes a conducting transistor that is connected to a fixed voltage corresponding to the second logic state.

According to an embodiment of the invention, the memory is a first-in/first-out (FIFO) memory that includes a first circular shift register whose outputs respectively control the read lines, the shifting of this register being controlled by the complement of the read selection signal; and a second circular shift register whose outputs respectively control the write lines, the shifting of this register being controlled by the complement of the write selection signal.

The foregoing and other objects, features, aspects and advantages of the invention will become apparent from the following detailed description of the present invention when taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF DRAWINGS

FIG. 1, above described, partially represents a conventional dual-port memory structure;

FIG. 2 represents the waveform of various signals of the memory of FIG. 1 during read cycles;

Fig. 3 partially represents an embodiment of a dual-port memory structure according to the invention;

FIG. 4 represents the waveforms of various signals of the memory of FIG. 3 during a read cycle;

FIG. 5 represents an embodiment according to the invention of a control circuit for controlling the read and write lines of a FIFO memory; and

FIG. 6 represents an exemplary circuit for generating state signals of a FIFO memory provided with the control circuit of FIG. 5.

DETAILED DESCRIPTION

In FIG. 3, same elements as in FIG. 1 are labeled with same reference numerals.

The invention provides a dummy memory cell **10'** for all of the columns of cells; the output of this dummy cell is connected to a dummy output line Bo' through an N-channel MOS transistor M2'. The output of this dummy cell **10'** is constantly forced to "0" through an N-channel MOS transistor **10-2** that is connected between transistor M2' and ground, and whose gate is connected to voltage Vdd. Transistors M2' and **10-2** are selected so as to have the same characteristics, respectively, as transistors M2 and the N-channel output transistor of a memory cell.

The dummy output line Bo' is connected to voltage Vdd, as the other output lines Bo, by a precharge transistor MP1 whose gate is controlled by the read-column signal RC. In contrast to the read transistors M2, the gate of the read transistor M2' of the dummy cell is connected to line RC. With this configuration, the dummy cell **10'** is "read" simultaneously with any one of the other "real" cells, and the dummy output line Bo' behaves exactly in the same way as line Bo of the real cell that is read.

The gates of the output transistors M3, instead of being connected to line RC, are connected to a line RC2 that is connected to the output of an AND gate **16**. A first input of the AND gate **16** is connected to line RC and the second input is connected to the dummy output line Bo' through an inverter **18**.

FIG. 4 shows the waveforms of various signals of the memory of FIG. 3 during the read cycle of a "1" in a memory cell whereas a "1" was previously read in a cell of the same column. This case corresponds to time t3 of FIG. 2 where the output of a conventional memory undesirably passed through "0".

At time t3, lines R and RC are asserted. The precharge transistors MP1 are off; transistor M2' and transistor M2 of the cell to be read turn on. The output of the dummy cell **10'** and the output of cell **10** to be read are at "0", which causes, as shown, the output lines Bo and Bo' to be progressively and synchronously discharged. During a first discharge phase of line Bo', the threshold voltage Vt of inverter **18** is not reached; the output of gate **16** remains at "0" and the output transistors M3 remain off. This first phase corresponds to the time interval during which the output of a conventional memory undesirably passes through "0", as represented in dotted lines for the output Bo1 . This undesirable passage through "0" is avoided, according to the invention, due to the fact that the output transistors M3 remain off during this first phase.

When the state of the dummy output line Bo' reaches the threshold voltage Vt, the output of inverter **18** switches to "1". The two inputs of gate **16** are then at "1", which causes line RC2 to be asserted. The output transistors M3 turn on, but since the state of line Bo (that varies as that of line Bo') is below the threshold voltage Vt, latch 12 (whose input was at "0") does not switch. Hence, output Bo1 continuously remains at "1".

When lines R and RC are disabled, line RC2 is immediately disabled, and lines Bo and Bo' are again charged to voltage Vdd.

With this configuration, two transitions, corresponding to a short undesired passage through "0", of each memory output during two consecutive read cycles of a "1", are avoided. Of course, the addition of a dummy cell **10'** and of a dummy output line Bo' increases current consumption. This current consumption is however low as compared with the consumption caused by one undesired passage through "0" at a memory output, because the memory outputs are connected to lines having a significant length.

3. The dual-port memory of claim 2, wherein said means for turning on the output transistors include a logic gate having a first input connected to the dummy output line through an inverter and a second input that receives the read selection signal.

4. The dual-port memory of claim 2, wherein said dummy memory cell includes a conducting transistor connected to a fixed voltage corresponding to the second logic state.

5. A circuit for controlling a write cycle of a memory device, the memory device having input column lines, at least one memory cell coupled to each input column line, a plurality of data lines, and at least one write line that is respectively coupled to the at least one memory cell, the controlling circuit including a plurality of input transistors, each input transistor having a first terminal respectively connected to one of the plurality of data lines, a second terminal respectively coupled to one of the input column lines, and a third terminal coupled to a write-column line;

wherein the input transistors are all simultaneously activated by the write-column line which is simultaneously activated when any one of the at least one write line is activated.

6. A circuit for controlling a write cycle of a memory device, the memory device having input column lines, at least one memory cell coupled to each input column line, at least one write line that is respectively coupled to the at least one memory cell, and a plurality of data lines, the circuit comprising:

a write-column line, coupled to each of the at least one write line, the write-column line being simultaneously activated when any one of the at least one write line is activated: and

means, connected between the input column lines, and the data lines, for simultaneously decoupling each of the plurality of data lines from each of the respective input column lines whenever the write-column line is not activated.

7. A memory device, comprising:

input column lines;

at least one memory cell coupled to each input column line;

a plurality of data lines;

at least one write line that is respectively coupled to the at least one memory cell; and

a plurality of input transistors, each input transistor having a first terminal connected to a respective one of the plurality of data lines, a second terminal coupled to a respective one of the input column lines, and a third terminal coupled to a write-column line;

wherein the input transistors are all simultaneously activated by the write-column line which is simultaneously activated when any one of the at least one write line is activated.

* * * * *

US006025746A

United States Patent [19]

So

[11] Patent Number: 6,025,746

[45] Date of Patent: Feb. 15, 2000

[54] **ESD PROTECTION CIRCUITS**

[75] Inventor: **Jason Siucheong So**, Carrollton, Tex.

[73] Assignee: **STMicroelectronics, Inc.**, Carrollton, Tex.

[21] Appl. No.: **08/771,670**

[22] Filed: **Dec. 23, 1996**

[51] **Int. Cl.**[7] **H02H 9/04**

[52] **U.S. Cl.** **327/325**; 327/314; 327/328; 361/56; 361/91

[58] **Field of Search** 327/313, 314, 327/320, 325, 327, 328; 257/173, 355; 361/56, 91

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,401,905	8/1983	Conger	327/325
5,343,352	8/1994	Nagamine	361/56
5,579,200	11/1996	Rajkanan et al.	361/111
5,651,574	7/1997	Williams et al.	257/355
5,672,527	9/1997	Lee	437/41 RLD
5,793,588	8/1998	Jeong	361/56

FOREIGN PATENT DOCUMENTS

2308741 7/1997 United Kingdom .

OTHER PUBLICATIONS

Sung-Mo Kang and Yusuf Leblebici, "CMOS Digital Integrated Circuits: Analysis & Design", McGraw-Hill, 1995, pp. 480–483.

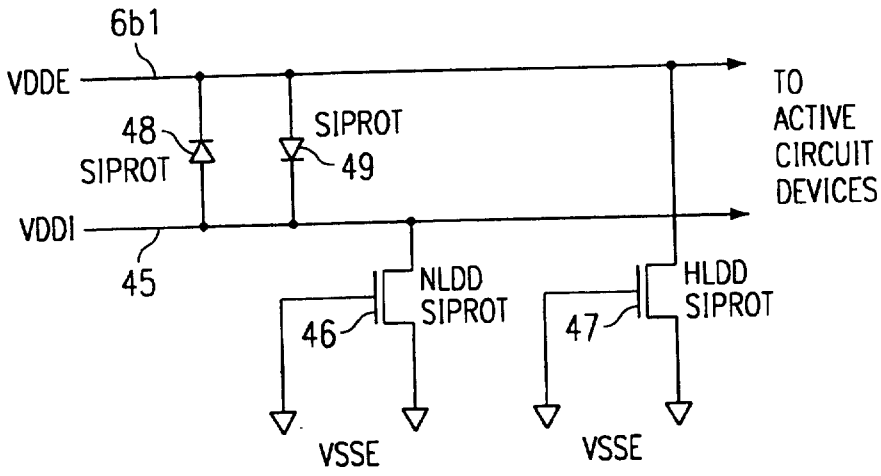
Primary Examiner—Terry D. Cunningham

Attorney, Agent, or Firm—Theodore E. Galanthay; Lisa K. Jorgenson

[57] **ABSTRACT**

Electro-static-discharge (ESD) protection circuits are supplied for inhibiting the destruction of buffers, drivers, logic and memory cells in Metal-Oxide-Semiconductor (MOS) devices such as a CMOS device including Static-Random-Access-Memory (SRAM). This is accomplished by tiering diodes adjacent the input of the chip and in certain specific areas internally of the chip (e.g. power supplies etc.) providing bidirectional diode protection from over-voltage.

5 Claims, 4 Drawing Sheets



Ex E

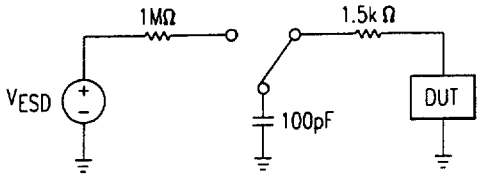


FIG. 1A
(PRIOR ART)

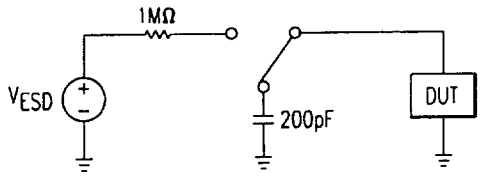


FIG. 1B
(PRIOR ART)

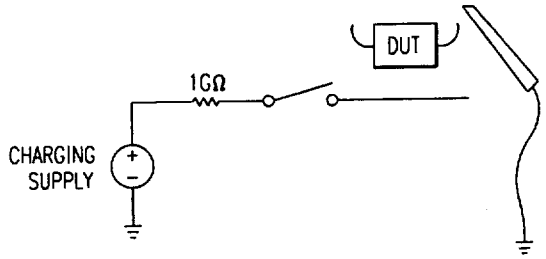


FIG. 1C
(PRIOR ART)

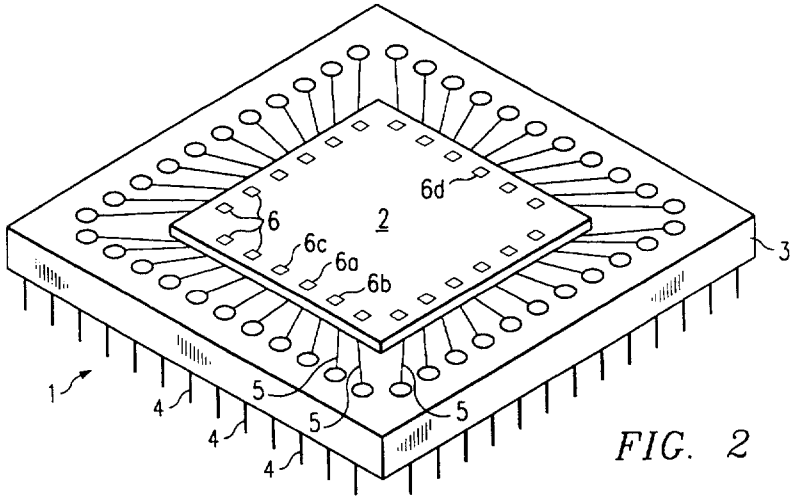


FIG. 2

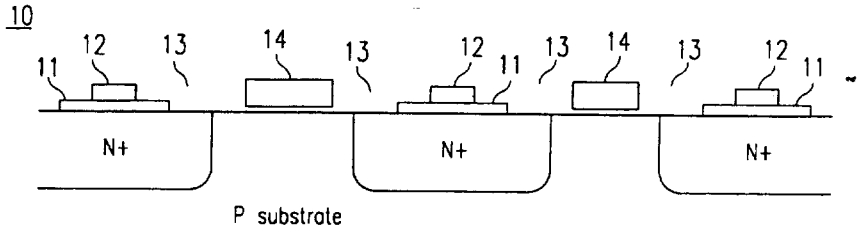


FIG. 3A

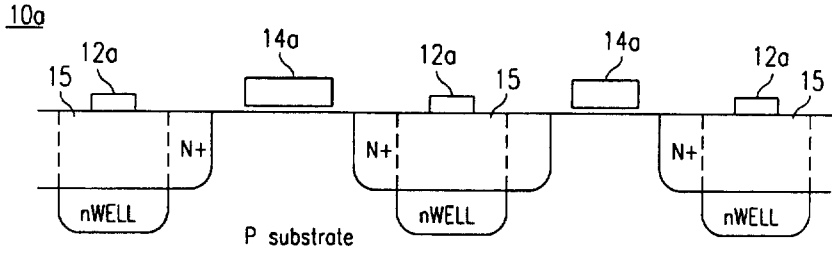


FIG. 3B

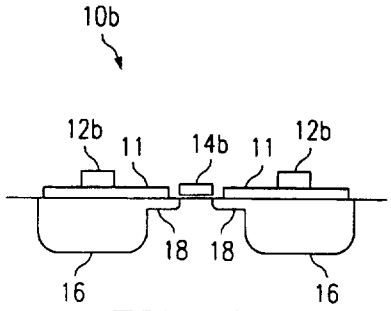


FIG. 3C

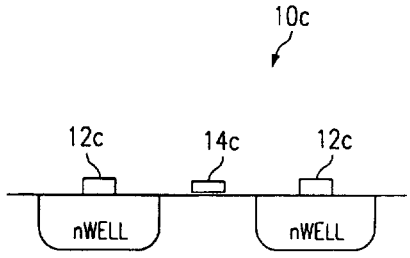


FIG. 3D

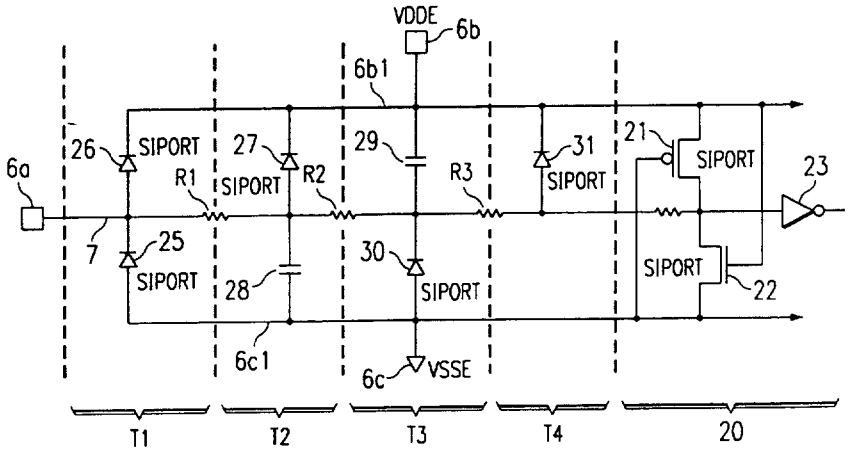

VDDE
6b
6b1
6a
26
SIPORT
R1
27
SIPORT
R2
29
R3
31
SIPORT
21
SIPORT
23
7
25
SIPORT
28
30
SIPORT
SIPORT
22
6c1
6c
VSSE
T1
T2
T3
T4
20

FIG. 4

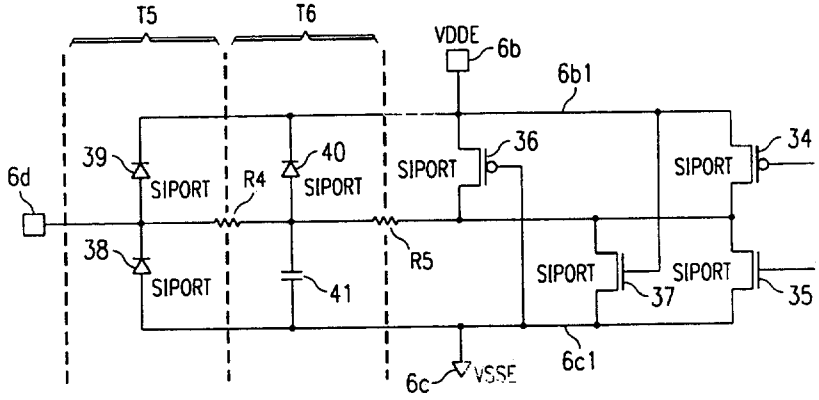

T5
T6
VDDE
6b
6b1
6d
39
SIPORT
R4
40
SIPORT
SIPORT
36
SIPORT
34
38
SIPORT
41
R5
SIPORT
37
SIPORT
35
6c
VSSE
6c1

FIG. 5

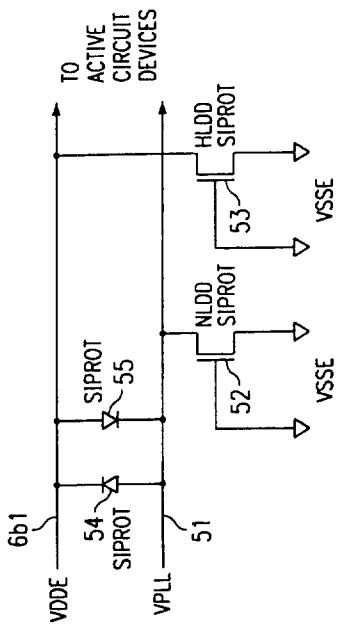


FIG. 7

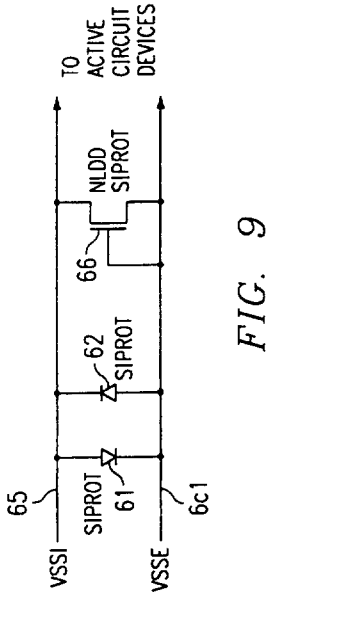


FIG. 9

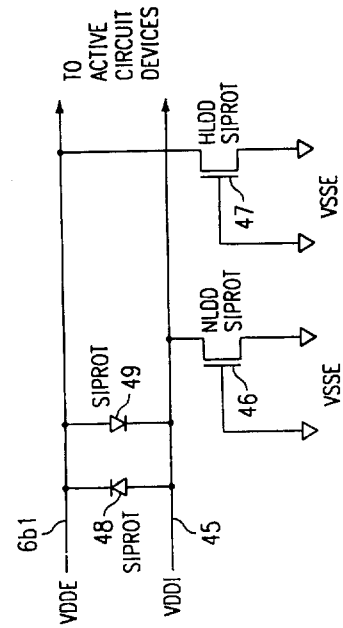


FIG. 6

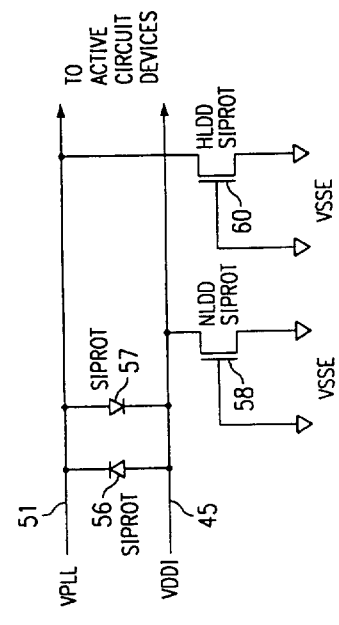


FIG. 8

ESD PROTECTION CIRCUITS

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to protection for Metal Oxide Semiconductor (MOS) integrated circuits such as Static-Random-Access-Memory (SCRAM) and more particularly relates to protection schemes against electrostatic discharge from either human or machine handling, while minimizing effects on the circuits to which the schemes are applied.

2. Description of Related Art

Electrostatic discharge (ESD) is one of the most prevalent causes for chip failures in both chip manufacturing and field operations. ESD can occur when the charges stored in machines or the human body are discharged to the chip on contact or by static induction. FIG. 1 shows different models for ESD testing. FIG. 1A shows the test for duplicating the human body model (HBM); FIG. 1B the test for duplicating the machine model (MM), and; FIG. 1C the test for duplicating the Charged Device Model (CDM).

A human walking across synthetic carpet in 80% relative humidity can potentially induce 1.5 kV of static voltage stress. In the HBM (MIL-STD 883C; Method 3015. 1988) shown in FIG. 1A, a touch of a charged person's finger is simulated by discharging a 100-pF capacitor through a 1.5K resistor. It is important that some protection network be designed into the I/O circuits of the chip so that the ESD effect can be filtered out before its propagation to the internal logic circuit effects destruction of one or more circuit elements. In addition to human handling, contact with machines can also cause ESD stress. Since body resistance is absent, the stress can be even more severe and with higher current levels. The schematic diagram of the machine model is shown in FIG. 1B. In that model the 1.5K resistor, representing the human in line impedance, is removed and a straight short condition with a 200 pf capacitor is discharged directly through the grounded DUT (device under test).

The third model is the charged device model shown in FIG. 1C. This model is intended to illustrate the discharge of the packaged integrated circuits. The charge can be accumulated either during the chip assembly process or in the shipping tubes. The CDM ESD testers electrically charge the Device Under Test (DUT) and then discharge it to ground, thus providing the high short-duration current pulse to DUT.

Small chip size, small diameter power bussing metal runs, and limited power/ground pins present particular problems with ESD irrespective of the type of CMOS device.

Larger chips have more capacitance (more available for charge storage) and are easier to handle; the smaller the chip, the more difficult the ESD problem. Moreover, narrow power bussing metal runs means the current carrying capacity of the metal is limited (acts like a fuse) and ofttimes, a static discharge will blow (melt to short or open) the line.

Power buss size is a dependency, requiring an ESD solutions. In the same manner limited ground line area is an obvious problem, because the line cannot dissipate heat due to excess current fast enough and sometimes causes adjacent device breakdown.

SUMMARY OF THE INVENTION

In view of the above, it is a principal object of the present invention to provide means for nullifying, to the extent possible, the deleterious effects of electro-static discharge in MOS devices.

Another object of the present invention is to provide means for enhancing the discharge of electrostatic charges on chips by providing an easier path to ground and bypassing the active and passive elements to ground.

The forgoing objects are accomplished by Electrostatic-discharge (ESD) protection means which include tiered diodes and capacitors adjacent the input and output pins of the chip. Additionally, bi-directional diode protection from over-voltage is provided in certain specific internal areas of the chip to protect against internal shorts to and from the power supplies.

Other objects and a more complete understanding of the invention may be had by referring to the following description taken in conjunction with the accompanying drawings in which:

BRIEF DESCRIPTION OF THE DRAWING(S)

FIG. 1A is a schematic, prior art diagram showing the ESD test for duplicating the human body model (HBM).

FIG. 1B is a schematic, prior art diagram showing the ESD test for duplicating the machine model (MM).

FIG. 1C is a schematic, prior art diagram showing the ESD test for duplicating the charged device model (CDM).

FIG. 2 is a perspective view of an example IC package with an integrated circuit MOS device thereon including protection means constructed in accordance with the present invention.

FIG. 3A is a fragmentary, cross sectional view of a portion of a CMOS integrated circuit chip and an n-channel transistor in which salicide is employed in conjunction with the chip, and openings are provided therein surrounding the N+ contacts of the active devices so that nWELLs are unnecessary.

FIG. 3B is a fragmentary, cross sectional view of a portion of a CMOS integrated circuit chip and n-channel transistor utilizing no salicide but employing nWELLs below N+ contacts;

FIG. 3C is a fragmentary, cross sectional view of a p or n channel transistor employing LDD and salicide;

FIG. 3D is a fragmentary, cross sectional view of a p or n channel transistor with NLDD and processed without salicide, necessitating nWELLs;

FIG. 4 is a schematic diagram illustrating a first protection scheme for input in a MOS integrated circuit device;

FIG. 5 is a schematic diagram illustrating a second protection scheme for outputs in a MOS integrated circuit device;

FIGS. 6–9 are schematic diagrams illustrating protection schemes specifically tailored for protection of power supplies internally of an MOS integrated circuit device.

DESCRIPTION OF THE ILLUSTRATIVE EMBODIMENT(S)

Turning now to the drawings, and specifically FIG. 2 thereof, an exemplary integrated circuit (IC) package 1 is illustrated as including an integrated circuit MOS device 2 thereon including ESD protection means constructed in accordance with the present invention. The package 1 includes an insulating platform 3 from which pins 4 depend from the lower surface thereof. The pins 4 pass through the platform and connect to leads 5, which connect in turn to input/output pads 6 on the MOS IC chip 2. While certain of the protection means may be incorporated externally of the chip 2 on the insulating platform 3, as shall be illustrated

hereinafter, the ESD protection means is preferably located in the device **2** so that protection will include the handling of the device for securing it to the platform **3** of the package **1**. In this connection, the present invention is not meant to exclude other ESD protection which may be imposed by the packaging of the integrated circuit **2**.

Prior to discussing the elements of the ESD protection means of the present invention, some background concerning the chip or device is essential to proper understanding of the workings of the protection schemes. Turning now to FIG. **3**A, a fragmentary, cross sectional view of a portion of a MOS integrated circuit chip **10** and an n-channel transistor in which salicide **11** is employed in conjunction with the chip **10**, underlying the N+ contacts **12**. Openings **13** are provided therein surrounding the N+ contacts of the active device so that nWELLs are unnecessary.

Salicide is a silicon compound with titanium in the compound to form into a conductive layer. The openings **13** are formed in any convenient manner, masking, etching etc and often are referred to as "salicide protected" areas. Elements that are in the salicide protected areas are referred to as SIPROT elements.

The chip substrate is composed of p-doped silicon. Intermediate the contacts **12** is the polysilicon gate **14**. The N+ region is below and in electrical contact with contacts **12** through the conductive salicide layer **11**. The lack of nWELLs in the N+ area enhances the turn-on of diodes and parasitic diodes.

In FIG. **3**B, a fragmentary, cross sectional view of a portion of a MOS integrated circuit chip **10***a*. Unlike the chip **10** shown in FIG. **3**A, chip **10***a* has no salicide layers. Without salicide, nWELLs are necessary for underlying the N+ contacts **12***a* to inhibit punchthrough.

In FIG. **3**C, a fragmentary, cross sectional view of a p-channel or n-channel transistor **10***b* employing a salicide layer **11** intermediate the contacts **12***b* and the appropriately doped regions **16** for the contacts is shown. An LDD **18** area (lightly doped drain-source diffusion) is also shown. The polysilicon gate **14***b* is also shown intermediate the contacts **12***b* and salicide layers **11**. The gate **14***b*, like the gate **14** shown in FIG. **3**A is SIPROT (salicide protected).

FIG. **3**D illustrates a fragmentary, cross sectional view of a p-channel or n-channel transistor **10***c* with NLDD (No Lightly Doped Drain-source Diffusion) and processed without salicide, necessitating nWELLs for the contacts **12***c*. The polysilicon gate **14***c* is illustrated intermediate the contacts **12***c*.

In the following discussion, the protection means discussed are all considered to be salicide protect or (SIPROT) which means that they are protected from salicide and also the protection means is not LDD (NLDD).

Turning now to FIG. **4**, one of the input pads **6**, designated **6***a*, is coupled to an input line **7** leading to a buffer **20**, part of which is shown to the right in FIG. **4**. A pad **6***b* connects the external power supply voltage VDDE to a VDDE line or rail **6***b***1**, and a pad **6***c* connects the external power supply ground VSSE to a ground line or rail **6***c***1**. The buffer **20** includes a pMOS transistor **21**, the gate of which is connected to the external power supply ground VSSE through ground line **6***c***1**, and the drain of which is connected to the power supply voltage VDDE by way of the VDDE line or rail **6***b***1**. In a like manner, buffer **20** also includes an nMOS transistor **22**, the gate of which is connected to the external power supply voltage VDDE by way of the VDDE line or rail **6***b***1**, and the drain of which is connected to the power supply ground VSSE through ground line **6***c***1**. (This is the proper location of pull up/pull down resistors, if any are needed or desired for the circuit). The sources of both transistors **21** and **22** are connected to the input signal line **7**. The line **7** continues to an invertor-amplifier **23** and then continues to other logic and memory circuits (not shown).

In accordance with the invention, a plurality of protection diodes are arranged in tiers or stages **T1**–Tn on the device **2** intermediate at least the signal input line **7** and the external ground VSSE **6***c* via line **6***c***1** and an external power supply VDDE **6***b* via applied voltage line **6***b***1**. At least the first of the tiers T1 comprising at least a first diode **25** between the ground gel and signal line **7**, and a second diode **26** connected between the signal line **7** and supply voltage line **6***b***1**. As shown, the diodes **26** and **25** are normally biased in the cutoff mode so that without signal on the input pad **6***a*, input line **7** should float at about 2 VDDE.

As shown in FIG. **4**, at least a second tier **T2** includes a third diode **27** connected between the ground or power supply applied voltage line **6***b***1**, **6***c***1** and the signal line **7**. In the illustrated instance it is connected between the voltage supply line **6***b***1** and input line **7**. The semiconductor T2 also has at least one of a fourth diode and capacitor, in the illustrated instance a capacitor **28**, connected between the other of the ground or power supply applied voltage line **6***c***1**, **6***b***1** and the input signal line **7**. In the present instance, capacitor **28** is connected between the ground line **6***c***1** and the input line **7**. Intermediate the first and second tiers, T1 and T2, and in series with the signal line, is at least one low impedance resistor **R1**. It is preferred that this resistor be of low impedance, e.g. in the 100 ohm range, so as to offer small impedance to an input signal, but offer enough resistance to allow staging or stepping of electro-static discharge protection between the tiers.

In the third tier T3, a capacitor **29** is connected intermediate the external power supply VDDE **6***b* and the input signal line **7** and a reverse biased diode **30** is connected intermediate the signal line **7** and the external ground VSSE **6***c*. Connected between the second and third tiers, T2 and T3, and in series with the signal line **7**, is at least another low impedance resistor **R2**. Like resistor R1, it is preferred that this resistor be of low impedance, e.g. in the 100 ohm range, so as to also offer small impedance to an input signal, but offer enough resistance to allow staging or stepping of electro-static discharge protection between tiers T2 and T3.

If desired, a fourth stage or tier T4 may be provided including a reverse biased diode **31** connected between the input signal line **7** and the power supply rail or line **6***b***1**. In a like manner, tiers T3 and T4 are separated by a low impedance resistor R3.

In operation, assume that a person handling the chip **2** or integrated circuit package **1** touches the power supply pad **6***b* or pin associated therewith, and a charge is induced at pad **6***c*. The least impedance path to the pad **6***b* is through tier T1 and the series connected diodes **25** and **26**, allowing the charge to dissipate through the human to ground. The higher impedance paths of T2 and T3 (due to the capacitors **28** and **29**), while carrying some smaller portion of the electro-static discharge, will not carry the full charge. However, if the diodes **25** and **26** cannot carry all the current, the excess current will be shunted through R1 into tier T2 and diode **27** to pad **6***b* and the human making contact. The rationale for

the low value of resistor R1 now becomes apparent. Even though a small value, the instantaneous current passing through the resistor R1 causes a voltage drop and a lowering of the impressed voltage upon diode 27. The same analysis holds true for tier T3 and the charging capacitor 29 which acts as a voltage charge buffer and does not discharge until the voltage at VDDE (pad 6*b*) drops below the capacitor charge voltage. Once again, the resistor R2 serves the same purpose as resistor R1, except that the impressed voltage and the current passing along signal line 7 is once again reduced or staged. In the unlikely event that the charge is still not dissipated in the manner described above, diode 31 in tier T4 will allow any remaining charge to be dissipated through resistor R3, diode 31 and to the pad 6*b*.

In the event that the human is touching input pad 6*a*, the analysis proceeds as before from an applied charge to pad 6*c* (VSSE), except that the diode 25 will act as the first line of defense for shunting the charge into signal line 7 and to the input pad 6*a*.

While the kind of staged protection of inputs and external to internal lines and leads described above is desirable, output lines also require a measure of protection from electro-static discharge. To this end and referring now to FIG. 5, illustrated therein is a similar form of protection to that already discussed relative to FIG. 4. Once again, a tiered protection scheme is displayed. In the right hand portion of FIG. 5 are output driver p-channel transistor 34 and n-channel transistor 35. The transistors are series connected between the power supply line 6*b*1 and the ground line 6*c*1. The output line 8 is connected to the interconnection between the two transistors. P-channel transistor 36 and n-channel transistor 37 act as part of the buffers and are also the location for pull-up and pull-down functions as necessary. As shown, the output line 8 leads to output pad 6*d* on the chip 2.

In accordance with a feature of the invention, the output circuitry is also protected from electro-static discharge by tiering the protection. To this end, reverse biased, series connected diodes 38 and 39 are connected between the rails 6*b*1 and 6*c*1 (VDDE and VSSE) and the output line 8 and thus the output pad 6*d*. As before, the first tier or stage protection T5 is separated from the second tier or stage T6 by a low impedance resistor R4. The second tier T6 is comprised of reverse biased diode 40, and capacitor 41. Diode 40 in the present instance is connected between the output line 8 and the rail 6*b*1 associated with the external power supply VDDE. Capacitor 41 is connected intermediate the output line 8 and the ground line 6*c*1.

While other tiers may be applied to the output, primarily dependent upon the ESD conditions where the package 1 is going to be used, it has been found that a two tier protection scheme plus separating the tiers from the buffers and output stages with a low impedance resistor, such as resistor R5 connected in series with R4 in the output line 8, is sufficient.

It should be recognized, that various parts of the integrated circuit MOS chip 2 have derived their power supply voltage from the external power supply voltage VDDE and ground from the external ground VSSE. Moreover, because of handling and the possibility of a charge being induced from any of the lines or leads, it is desirable to provide protection internally of the chip 2, between lines that are normally at the same potential. Just by way of example, the following table is provided and applies to all of the voltage acronyms set forth in the drawings:

Acronym	Explanation of Acronym	Typical Voltage
VDDE	Power supply, External	5 volts
VDDI	Power supply Internal	5 volts
VPLL	Power supply, Phase locked loop	5 volts
VSSE	Ground, External	0 volts
VSSI	Ground, Internal	0 volts

It is noted that the external power supply, VDDE, is normally the power supply for the peripheral transistors, i.e. the transistors surrounding the core or central portion of the chip. The internal power supply, VDDI, normally equal to the external power supply, is the power supply for the central logic in the chip.

Turning now to FIG. 6, bi-directional diode protection is provided between the internal and external voltage lines 45 and 6*b*1 for shunting static charges from one to the other, depending upon source. The protection means includes a pair of MOS transistors, in the present instance n-channel, transistors, 46 and 47, one connected between a line and external ground VSSE and the other connected between another line and the external ground VSSE with their gates connected to one or the other lines dependent upon transistor type. In the present instance, inasmuch as the transistors are nMOS or n-channel types, the gates are connected to VSSE. The bi-directional diodes 48 and 49 are normally turned off inasmuch as the voltage on both lines is the same.

In operation, assume a human is holding the chip 2 and contacting pad 6*b* for VDDE, and charge is applied to VDDI. Under these circumstances the first diode 48 will take the current and shunt the charge to the human, protecting active circuits and the like connected to the lines 6*b* 1 and line 45. If the opposite occurs, then the 2nd diode 49 will become the shunt path to human (assuming that it is possible for a human to touch VDDI).

Alternatively, if the person is holding ground pin VSSE, and a charge is applied through VDDI, then the first transistor 46 will shunt the current through that transistor 46 to the person. The same is true of a charge coming from VDDE, it will pass through second nMos transistor 47 to VSSE and thus to the human thereby protecting circuitry attached to one or both of lines 6*b*1 and 45.

The scheme shown in FIG. 7 is identical in operation to the scheme employed in FIG. 6. In FIG. 7, protection is sought between the external power supply VDDE and internal power supply line 51 for VPPL and external ground VSSE. The protection means includes a pair of MOS transistors, in the present instance n-channel, transistors, 52 and 53, one connected between the one line and external ground VSSE and the other connected between the other line and the external ground VSSE with their gates connected to one or the other lines dependent upon transistor type. In the present instance, inasmuch as the transistors are nMOS or n-channel types, the gates are connected to VSSE. The bi-directional diodes 54 and 55 are normally turned off inasmuch as the voltage on both lines is the same.

In operation, assume a human is holding the chip 2 and contacting pad 6*b* for VDDE, and a charge is applied to VPPL. Under these circumstances the first diode 54 will take the current and shunt the charge to the human, protecting active circuits and the like connected to the lines 6*b*1 and line 51. If the opposite occurs, then the 2nd diode 55 will

become the shunt path to a human (assuming that it is possible for a human to touch VPLL).

Alternatively, if the person is holding ground pin for VSSE, and a charge is applied through VPPL, then the first transistor **52** will shunt the current through that transistor to the person. The same is true of a charge coming from VDDE, it will pass thru second nMOS transistor **53** to VSSE and thus to the human, protecting circuitry attached to one or both or lines **6***b***1** and **51**.

Another protective scheme may be employed between the power line **51** for phase locked loop, VPPL, and the internal power supply line **45** (VDDI) from which it is derived. In this scheme, illustrated in FIG. **8**, a pair of bidirectional diodes **56** and **57** will allow for shunting of electro-static discharge through nMOS transistors **58** and **60** to ground VSSE to pad **6***c*.

In FIG. **9**, additional protection for electrostatic discharge through ground, is provided. In this scheme, bidirectional diodes **61** and **62** are provided intermediate an internal ground line **65** (VSSI) and external ground power supply line **6***c***1** (VSSE). In the illustrated instance, nMOS transistor **66** is connected in parallel between the ground lines to facilitate quicker turn on and discharge of an ES discharge before it reaches the active circuits, and devices connected to the internal and external ground lines. Although not shown, similar protection may be applied to the ground associated with the phase locked loop circuitry.

Thus, the present invention provides means for nullifying, to the extent possible, the deleterious effects of electro-static discharge in MOS devices. Moreover, the ES protection circuits of the present invention enhances the discharge of electro-static charges on chips by providing an easier path to ground bypassing threatened active and passive elements to ground. This is accomplished in the present invention by Electro-static-discharge (ESD) protection circuits which include tiered diodes and capacitors adjacent the input and output pads of the chip. In certain specific internal areas of the chip, to protect against internal shorts to and from the power supplies, bi-directional diode protection from over-voltage caused by ESD is provided.

Although the invention has been described with a certain degree of particularity, it should be recognized that elements thereof may be altered by person(s) skilled in the art without departing from the spirit and scope of the invention as hereinafter set forth in the following claims.

What is claimed is:

1. In a MOS integrated circuit semiconductor device having electrostatic protection circuits protecting active and passive components in the device from electrostatic discharge on an external power supply voltage line, and wherein said device includes an internal power supply voltage line whose voltage is derived from the external power supply voltage line, additional protection circuitry in the device for protecting the active and passive components from electrostatic discharge between an external power supply line and an internal power supply line, said additional protection circuitry comprising:

bi-directional diodes between said internal and external voltage lines for shunting electrostatic charges between said internal voltage line to said external voltage line;

and a pair of MOS transistors, one transistor connected between said internal voltage line and a ground line and the other transistor connected between the ground line and said external voltage line, the gate of each transistor connected to one of the ground line or to the transistor's connected voltage line depending upon whether the transistor is p-type or n-type, said transistors shunting an electrostatic charge between the ground line and one or the other of the voltage lines.

2. The additional electrostatic protection circuitry of claim **1** wherein said pair of MOS transistors are nMOS, and NLDD and SIPROT.

3. The additional electrostatic protection circuitry of claim **1** wherein said internal power supply voltage line is a phase-locked loop power supply voltage line.

4. The additional electrostatic protection circuitry of claim **3** wherein said pair of MOS transistors are nMOS, NLDD and SIPROT.

5. In a MOS integrated circuit semiconductor device having electrostatic protection circuits protecting active and passive components in the device from electrostatic discharge on an external power supply voltage line, and wherein said device includes at least two internal power supply voltage lines whose voltages are derived from the external power supply voltage line, additional protection circuitry in the device for protecting the active and passive components from electrostatic discharge between the internal power supply lines, said additional protection circuitry comprising:

bi-directional diodes between said internal voltage lines for shunting electrostatic charges between said internal voltage lines;

and a pair of MOS transistors, each transistor connected between one of said internal voltage lines and a ground line, the gate of each transistor connected to one of the ground line or to the transistor's connected internal voltage line depending upon whether the transistor is p-type or n-type, said transistors shunting an electrostatic charge between the ground line the transistor's connected internal voltage line.

* * * * *

US006087709A

United States Patent [19]

Gandy et al.

[11] Patent Number: 6,087,709

[45] Date of Patent: Jul. 11, 2000

[54] **METHOD OF FORMING AN INTEGRATED CIRCUIT HAVING SPACER AFTER SHALLOW TRENCH FILL AND INTEGRATED CIRCUIT FORMED THEREBY**

[75] Inventors: **Todd Gandy**, Phoenix; **Ronald Sampson**, Fountain Hills; **Robert Hodges**, Phoenix, all of Ariz.

[73] Assignee: **STMicroelectronics, Inc.**, Carrollton, Tex.

[21] Appl. No.: **09/379,391**

[22] Filed: **Aug. 23, 1999**

Related U.S. Application Data

[62] Division of application No. 08/996,457, Dec. 23, 1997.

[51] **Int. Cl.**[7] **H01L 29/06**

[52] **U.S. Cl.** **257/626**; 257/622; 257/506

[58] **Field of Search** 257/647, 649, 257/506, 622, 626, 510; 438/221, 296, 359

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,297,694	10/1981	Matherat	340/750
4,368,436	1/1983	Palara et al.	330/207
4,547,793	10/1985	Bergeron	257/510
4,700,464	10/1987	Okada et al.	437/67
4,808,550	2/1989	Fukushima	437/67
4,868,136	9/1989	Ravaglia	437/38
4,891,788	1/1990	Kreifels	365/49
5,011,788	4/1991	Kawaji et al.	437/67
5,130,268	7/1992	Liou et al.	437/67
5,244,827	9/1993	Dixit et al.	437/70
5,247,481	9/1993	Conan	365/200
5,355,341	10/1994	Gaultier et al.	365/201
5,395,785	3/1995	Nguyen et al.	437/52
5,410,176	4/1995	Liou et al.	257/50
5,548,554	8/1996	Pascucci et al.	365/200
5,729,043	3/1998	Shepard	257/519
5,763,315	6/1998	Benedict et al.	438/424
5,930,620	7/1999	Wristers et al.	438/243
5,952,707	9/1999	Hodges	257/640
5,960,297	9/1999	Saki	438/424
5,970,363	10/1999	Kepler et al.	438/435

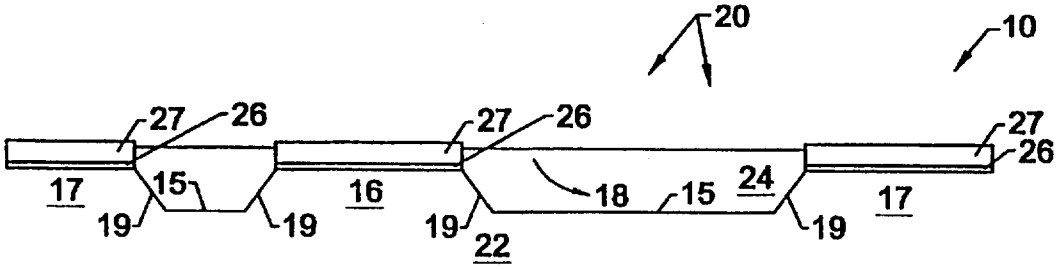
Primary Examiner—David Hardy

Attorney, Agent, or Firm—Theodore E. Galanthay; Lisa K. Jorgenson; Christopher F. Regan

[57] **ABSTRACT**

A method of forming an isolation region in an integrated circuit and an integrated circuit formed thereby. A method preferably includes forming at least one trench in a semiconductor substrate, forming an insulation layer of material in the at least one trench and on peripheral regions of the at least one trench of the semiconductor substrate, forming a sacrificial layer of material on the insulation layer having a different polishing rate than the insulation layer, and polishing the layer having the different polishing rate and portions of the insulation layer so that the sacrificial layer having the different polishing rate and portions of the insulation layer are removed, so that other portions of the insulation layer remain in the at least one trench of the substrate, and so that the upper surface of the at least one trench and the peripheral regions thereof in combination provide a substantially planar surface.

7 Claims, 1 Drawing Sheet



Ex F

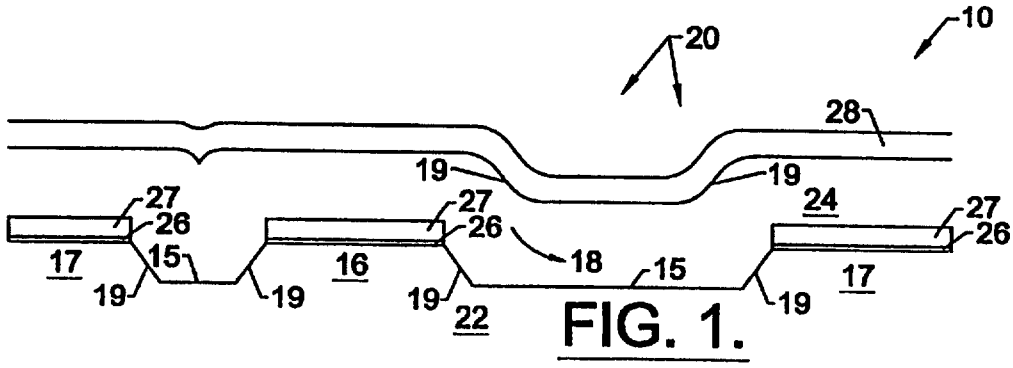


FIG. 1.

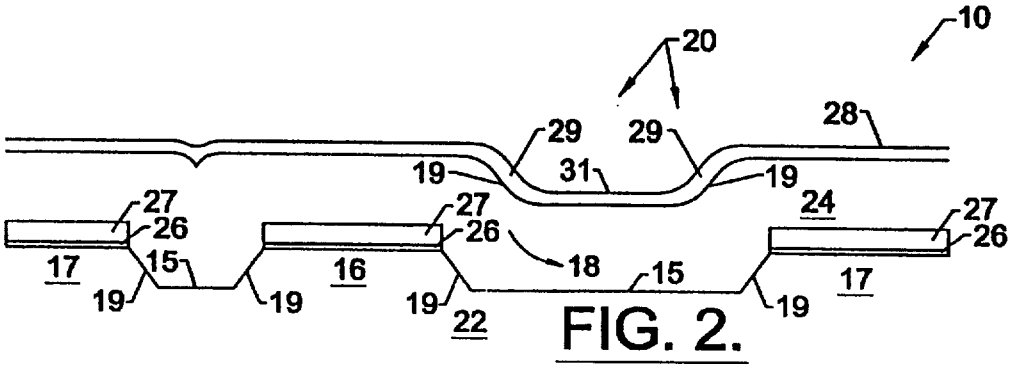


FIG. 2.

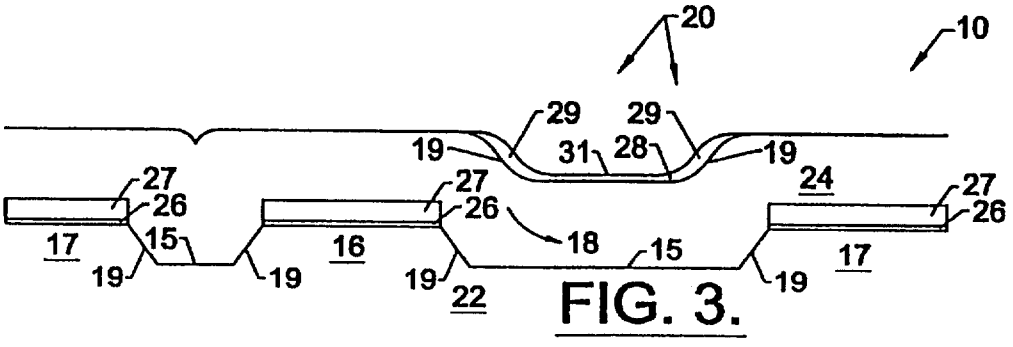


FIG. 3.

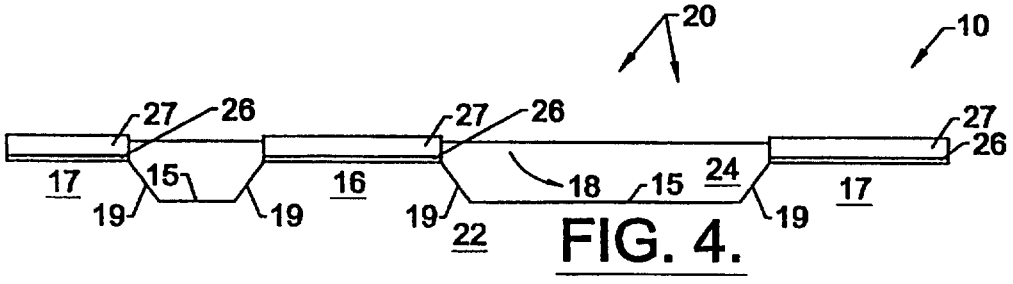


FIG. 4.

METHOD OF FORMING AN INTEGRATED CIRCUIT HAVING SPACER AFTER SHALLOW TRENCH FILL AND INTEGRATED CIRCUIT FORMED THEREBY

This application is a division of Ser. No. 08/996.457 filed on Dec. 23, 1997.

FIELD OF THE INVENTION

The invention relates to the field of integrated circuits, and, more particularly, to the field of integrated circuit fabrication.

BACKGROUND OF THE INVENTION

Over the years, in the field of integrated circuits, it has been recognized that the manufacturing cost of a given integrated circuit is largely dependent upon the chip area required for implementation of a desired function. Accordingly, the geometries and sizes of active components, such as the gate electrode in metal-oxide-semiconductor ("MOS") technology, are important elements in defining the chip area for a given integrated circuit. These geometries and sizes, in turn, are often dependent upon the photolithographic resolution available for a particular manufacturing facility.

As the degree of integration has advanced over the years, however, it has been recognized that it is desirable to minimize the topographical excursion of the surface at each level, especially the upper levels. To accomplish this, various planarization techniques have been developed to planarize the interlevel dielectric. Some of these, for example, include chemical-mechanical-polishing ("CMP"), use of permanent spin-on-glass ("SOG"), e.g., left in place in the final chip, and sacrificial etchback SOG.

SOG deposition has been used in the semiconductor industry for many years. The unprocessed SOG material is a fluid or gel-like material. After the fluid material is coated onto the face of a wafer, the wafer is rotated at high speed to throw off or discard the excess material. The surface tension and adhesion of the material provides a generally flat or substantially planarized surface with a controlled thickness. The fluid material is then baked in order to drive off solvents and provide a stable solid silicate glass.

An example of pre-metal planarization can be seen in U.S. Pat. No. 5,395,785 by Nguyen et al. titled SRAM Cell Fabrication With Interlevel Dielectric Planarization." This patent describes a method wherein planarization of a static random access memory ("SRAM") cell is performed before metal formation and before resistor formation. The pre-metal planarization utilizes a sandwich structure having permanent SOG, undoped glass, and permanent SOG. The undoped glass is used as a buffer layer between two layers of SOG to prevent SOG cracks. The double SOG enhances the degree of planarization.

Another factor in the chip area required for integrated circuits is the isolation technology. Sufficient electrical isolation must be provided between active circuit elements so that leakage therebetween does not cause functional or specification failure. The isolation is particularly important for circuits such as SRAMs where maintenance of stored data by extremely low levels of standby current has become highly desirable. Because the presence of leakage between active regions in the memory will greatly increase the standby current drawn, such low standby currents require excellent isolation. This increasingly difficult standby current requirement or desirability, in combination with the

demand for smaller and smaller memory cells in denser memory array, forms increased pressure on the isolation technology in SRAMs and other integrated circuits.

A known isolation technique, for example, is local oxidation of silicon ("LOCOS"). In LOCOS, an oxidation barrier is placed over the locations of the surface of the chip into which the active devices are to be formed, i.e., active regions. The wafer is then placed in an oxidizing environment. The portions of the wafer surface not covered by the oxidation barrier oxidize to form thermal silicon dioxide. Oxidation is masked from the active regions by the oxidation barrier. LOCOS field oxide is generally formed to a sufficient thickness that a conductor placed thereover will not invert the channel thereunder, when biased to the maximum circuit voltage. LOCOS, however, is subject to certain known limitations, including encroachment of the oxide into the active regions due to oxidation of silicon under the edges of the nitride mask and the adding of topography to the integrated surface.

A more recently known isolation technique uses trenches etched into the surface of the wafer at the isolation locations. The trenches are then filled with a thermal or deposited oxide. Such trench isolation can provide extremely thick isolation oxides which extend into the wafer surface with little or no encroachment. Etching of deep trenches, however, can be a relatively expensive process and can be quite difficult to perform when attempting to maintain close geometries. Also, thermally formed silicon dioxide, for example, generally has a higher integrity than deposited silicon dioxide. The formation of thermal silicon dioxide trenches, however, causes stress in the silicon due to volume expansion of silicon dioxide from that of the silicon prior to oxidation. Accordingly, trench isolation tends to largely rely on deposited oxide.

Even more recently, integrated circuits with planarized shallow trench isolation ("STI") have been developed such as seen in U.S. Pat. Nos. 5,130,268 and 5,410,176 each by Liou et al. and respectively titled Method For Forming Planarized Shallow Trench Isolation In An Integrated Circuit And A Structure Formed Thereby" and "Integrated Circuit With Planarized Shallow Trench Isolation." After formation of the recesses, sidewall filaments of insulating material, such as silicon dioxide, are formed into some or all of the recesses thereby exposing the bottom silicon portion thereof. Selective epitaxy then forms a silicon layer within the recesses from the bottom up, but not along the sides or sidewalls. The selective epitaxial layer is oxidized so that the recesses are substantially filled with thermal silicon dioxide. Both recesses with vertical sidewalls, formed by anisotropic etching of the silicon, or sloping sidewalls, formed by more isotropic silicon etching, can be utilized.

One of the key processes for STI is planarization after trench fill. The conventional methods generally require some type of inverse active pattern, resist/oxide etchback, resist strip, and CMP to complete the planarization process. These planarization process requirements, however, may not provide the consistent results desired. For example, the resulting integrated circuits using CMP for the planarization process can result in erosion of the corners extending along peripheral regions of the trench and can result in dishing along the upper surface of the filled trench.

SUMMARY OF THE INVENTION

In view of the foregoing background, the present invention advantageously provides a method of forming an integrated circuit having a substantially planar upper surface

substantially devoid of corner erosion and devoid of dishing. The present invention also advantageously provides a method of forming an integrated circuit having a spacer after shallow trench fill and an integrated circuit formed thereby which effectively eliminates conventional planarization process steps such as inverse active pattern and resist strip. The present invention additionally advantageously provides a method of forming an integrated circuit that reduces a likelihood of the presence of defects.

More particularly, the present invention provides a method of forming an isolation region in an integrated circuit which preferably includes the step of simultaneously polishing portions of an insulation layer on peripheral regions of at least one trench of a semiconductor substrate and a sacrificial layer of material overlying the at least one trench and having a different polishing rate than the insulation layer so that the sacrificial layer and portions of the insulation layer are removed, so that other portions of the insulation layer remain overlying the at least one trench of the substrate, and so that the upper surface of the at least one trench and the peripheral regions thereof in combination provide a substantially planar surface. The polishing, for example, is preferably chemical-mechanical-polishing ("CMP"). The sacrificial layer, e.g., a nitride layer, preferably has a slower polishing rate than the insulation layer, e.g., an oxide layer.

Another method of forming an isolation region in an integrated circuit according to the present invention preferably includes forming an insulation layer of material in at least one trench of a substrate and on peripheral regions of the at least one trench of the substrate and forming a sacrificial layer of material on the insulation layer having a different removal rate than the insulation layer. The method also preferably includes removing the layer having the different removal rate and portions of the insulation layer so that other portions of the insulation layer remain overlying the at least one trench of the substrate and so that the upper surface of the at least one trench and the peripheral regions thereof in combination provide a substantially planar surface.

An additional method of forming an isolation region in an integrated circuit according to the present invention preferably includes forming at least one trench in a semiconductor substrate, forming an insulation layer of material in the at least one trench and on peripheral regions of the at least one trench of the semiconductor substrate, and forming a sacrificial layer of material on the insulation layer having a different polishing rate than the insulation layer. The method preferably also includes polishing the layer having the different polishing rate and portions of the insulation layer so that the layer having the different polishing rate and portions of the insulation layer are removed, so that other portions of the insulation layer remain overlying the at least one trench of the substrate, and so that the upper surface of the at least one trench and the peripheral regions thereof in combination provide a substantially planar surface.

Yet another method of forming an isolation region in an integrated circuit preferably includes forming at least one trench in a surface of a semiconductor substrate so that the surface has at least two high regions and at least one low region thereof, forming a layer of oxide over the at least two high regions and the at least one low region of the substrate, forming a sacrificial layer of nitride on the oxide layer, and partially removing overlying portions of the nitride layer. The method preferably also includes removing the remaining portions of the nitride layer and portions of the oxide layer so that other portions of the oxide layer remain overlying the at least one low region of the substrate and so that the upper surface of the at least two high regions and the at least one filled trench in combination provide a substantially planar surface.

The present invention also provides an integrated circuit which includes a semiconductor substrate and at least one trench formed in the semiconductor substrate and which has a polished insulation layer substantially filled therein. The integrated circuit also preferably includes peripheral regions of the semiconductor substrate extending along the at least one trench of the semiconductor substrate so that the upper surface of the substantially filled at least one trench and the peripheral regions thereof in combination provide a substantially planar surface and so that the polished insulation layer substantially filling the at least one trench of the substrate is substantially devoid of dishing and so that corners of the peripheral regions of the at least one trench have relatively sharp edges and are substantially devoid of corner erosion.

A device for forming an isolation region of an integrated circuit is also provided according to the present invention. The device preferably includes a semiconductor substrate, at least one trench formed in the semiconductor substrate, and a conformal insulation layer on the at least one trench and peripheral regions of the at least one trench of the semiconductor substrate. The device preferably further includes a sacrificial layer of material on the conformal layer and overlying only the at least one trench. The sacrificial layer preferably has a different polishing rate than the conformal insulation layer and has a relatively thin layer thereof overlying medial portions of the at least one trench and a relatively thicker spacer over sidewall regions of the at least one trench.

BRIEF DESCRIPTION OF THE DRAWINGS

Some of the features, advantages, and benefits of the present invention having been stated, others will become apparent as the description proceeds when taken in conjunction with the accompanying drawings in which:

FIG. **1** is an enlarged sectional view of a process step of forming an isolation region of an integrated circuit according to the present invention;

FIG. **2** is an enlarged sectional view of another process step of forming an isolation region of an integrated circuit according to the present invention;

FIG. **3** is an enlarged sectional view of an additional process step of forming an isolation region of an integrated circuit according to the present invention; and

FIG. **4** is an enlarged sectional view of a further process step of forming an isolation region of an integrated circuit according to the present invention.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

The present invention will now be described more fully hereinafter with reference to the accompanying drawings, in which preferred embodiments of the invention are shown. This invention may, however, be embodied in many different forms and should not be construed as limited to the embodiments set forth herein. Rather, these embodiments are provided so that this disclosure will be thorough and complete, and will fully convey the scope of the invention to those skilled in the art. Like numbers refer to like elements throughout.

FIGS. **1–4** illustrate an integrated circuit structure or device **10** according to various process or method steps of

forming an isolation region **20** of an integrated circuit **10** of the present invention. FIG. **1**, for example, illustrates a semiconductor substrate **22** or wafer having relatively shallow trenches **15**, cavities, or recesses formed therein, e.g., STI. As such, these trenches **15** are relatively easy to etch without requiring high levels of anisotropy, high etch rates, or expensive etch equipment. As understood by those skilled in the art, the substrate **22**, for example, can be patterned, etched, and/or stripped prior to or along with the forming of the trenches **15**. Thin sidewalls can then be oxidized.

As understood by those skilled in the art, the present invention is applicable to single well, twin well, and triple well CMOS processes, and to other technologies including bipolar, n-channel and p-channel MOS, and BiCMOS technologies as well. Also, the conductivity type and concentrations may vary as well for these technologies. These technologies may have active devices formed directly into a monolithic substrate or into an epitaxial layer at the surface of the substrate. Accordingly, as understood by those skilled in the art, the present invention can be particularly beneficial to all CMOS types of devices or integrated circuits and, even more particularly, to sub-half micron CMOS type applications.

A method of forming an isolation region **20** in an integrated circuit **10** preferably includes forming at least one trench **15** in a surface of a semiconductor substrate **22** so that the surface has at least two high peripheral regions **16, 17** and at least one low region **18** thereof as illustrated in FIG. **1**. An insulation layer **24**, e.g., an oxide layer such as silicon dioxide, is formed, e.g., deposited, on the at least two high regions **16, 17** and on the at least one low region **18** of the substrate **22**. The at least two high regions **16, 17** of the semiconductor substrate **22**, for example, can include an oxide layer or pad **26** on a layer of silicon and a nitride layer or pad **27** on the oxide layer **26** positioned on the silicon layer. The forming of an insulation layer **24** of oxide on the semiconductor substrate **22** preferably includes forming the insulating oxide layer **24** over the nitride layer **27** of the at least two high regions **16, 17** of the substrate **22**.

A sacrificial layer **28** having a different removal or polishing rate is formed on the insulation layer **24**. The different polishing rate of the sacrificial layer **28** is preferably a slower polishing rate. The sacrificial layer **28**, for example, can include at least one of a nitride, titanium nitride, polysilicon, or an oxide having a different doping than the oxide of the insulation layer **24**. The insulation layer **24**, for example, can be a doped oxide and the sacrificial layer **28** can be an undoped oxide. The insulation layer **24**, however, preferably includes an oxide and the sacrificial layer **28** preferably includes a nitride. The nitride layer **28** preferably has a thickness in the range of 100–5000 Angstroms. In essence, the material or film can be any material having a different polishing rate and is preferably selectively chosen based upon the polishing or removal technique and the desired results.

As perhaps best illustrated in FIG. **2**, overlying or upper portions of the nitride layer **28** are partially removed. This partial removal of the nitride layer **28**, for example, can be performed by partially etching back the nitride layer **28** such as by using a plasma etch. As illustrated, this process leaves a relatively thin layer of nitride on the at least one high region **16, 17** and a relatively thicker nitride spacer **29** on or over sidewall regions **19** of the at least one trench **15**. Additionally, a relatively thin layer of nitride remains overlying medial portions of the conforming trench fill material.

As illustrated in FIG. **3**, the method can also include a partial polishing, e.g., CMP, of the nitride layer **28** so as to entirely remove portions of the nitride layer **28** only in the at least two high peripheral regions **16, 17** of the substrate **22**. Notably, relatively thick nitride spacers **29** remain along the sidewalls **19** of the conforming trench and a relatively thin nitride layer **31** remains on and overlies the medial portions of the conforming trench fill material.

FIG. **4** illustrates the removal of the remaining portions of the nitride layer **28** and portions of the oxide layer **24** such as by simultaneously polishing the remaining portions of the nitride layer **28** and portions of the oxide layer **24** so that the other portions of the oxide layer **24** positioned in the at least one low region of the substrate **22** are substantially devoid of dishing and so that corners of the at least two high regions of the substrate thereby form relatively sharp edges and are substantially devoid of corner erosion. In other words, the upper surface of the at least two high peripheral regions **16, 17** and the at least one filled trench in combination provide a substantially planar or flat surface.

As perhaps best illustrated in FIG. **4**, the present invention also provides an integrated circuit **10** which includes a semiconductor substrate **22** and at least one trench **15** formed in the semiconductor substrate **22** and which has a polished insulation layer **24** substantially filled therein. The integrated circuit **10** also preferably includes peripheral regions **16, 17** of the semiconductor substrate **22** extending along the at least one trench **15** of the semiconductor substrate **22** so that the upper surface of the substantially filled trench and the peripheral regions **16, 17** thereof in combination provide a substantially planar surface. Also, the polished insulation layer **24** which substantially fills the at least one trench **15** of the substrate **22** is also substantially devoid of dishing. Further, corners of the peripheral regions **16, 17** of the at least one trench **15** now advantageously have relatively sharp edges and are substantially devoid of corner erosion.

As perhaps best illustrated in FIG. **3**, the present invention also includes a device or integrated circuit structure **10** forming an isolation region **20** thereof. The device **10** preferably includes a semiconductor substrate **22**, at least one trench **15** formed in the semiconductor substrate **22**, and a conformal insulation layer **24** on the at least one trench **15** and peripheral regions **16, 17** of the at least one trench **15** of the semiconductor substrate **22**. The device **10** preferably further includes a sacrificial layer **28** of material on the conformal layer **24** and overlying only the at least one trench **15**. The sacrificial layer **28** preferably has a different polishing rate than the conformal insulation layer **24** and has a relatively thin layer thereof overlying medial portions of the at least one trench **15** and a relatively thicker spacer on or over sidewall regions **19** of the at least one trench **15**.

Many modifications and other embodiments of the invention will come to the mind of one skilled in the art having the benefit of the teachings presented in the foregoing descriptions and the associated drawings. Therefore, it is to be understood that the invention is not to be limited to the specific embodiments disclosed, and that modifications and embodiments are intended to be included within the scope of the appended claims.

That which is claimed:

1. An integrated circuit comprising:

a semiconductor substrate;

at least one trench formed in the semiconductor substrate so that the surface of the semiconductor substrate has at least two high regions and at least one low region, the at least one low region defining the at least one trench;

an insulation layer substantially filling the at least one trench, the insulation layer having a polished upper surface; and

the at least two high regions defining peripheral regions of the semiconductor substrate and extending along the peripheries of the at least one trench of the semiconductor substrate so that an upper surface of the substantially filled at least one trench and the peripheral regions thereof in combination provide a substantially planar surface and so that the polished upper surface of the insulation layer substantially filling the at least one trench of the substrate is substantially devoid of dishing and so that corners of the peripheral regions extending along the peripheries of the at least one trench have relatively sharp edges and are substantially devoid of corner erosion.

2. An integrated circuit as defined in claim **1**, wherein the insulation layer includes an oxide, and wherein the upper surface of each of the peripheral regions is also polished.

3. An integrated circuit as defined in claim **1**, wherein each of the at least two high regions includes an oxide layer on a layer of silicon and a nitride layer on the oxide layer positioned on the silicon layer.

4. A device for forming an isolation region of an integrated circuit comprising:

a semiconductor substrate;

at least one trench formed in the semiconductor substrate;

a conformal insulation layer on the at least one trench and peripheral regions of the at least one trench of the semiconductor substrate; and

a sacrificial layer of material on the conformal layer and overlying only the at least one trench, the sacrificial layer having a different polishing rate than the conformal insulation layer and having a relatively thin layer thereof overlying medial portions of the at least one trench and a relatively thicker spacer over sidewall regions of the at least one trench.

5. A device as defined in claim **4**, wherein the sacrificial layer having the different polishing rate has a slower polishing rate than the insulation layer.

6. A device as defined in claim **4**, wherein the insulation layer comprises a first oxide, and wherein the sacrificial layer having the slower polishing rate includes at least one of a nitride, a second differently doped oxide, polysilicon, and titanium nitride.

7. A device as defined in claim **5**, wherein the peripheral regions of the at least one trench includes at least two high regions of the semiconductor substrate, and wherein each of the at least two high regions includes an oxide layer on a layer of silicon and a nitride layer on the oxide layer positioned on the silicon layer.

* * * * *